VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

VIRTUÁLNÍ MODEL VÝROBNÍHO STROJE REALIZOVANÝ V PROSTŘEDÍ ABB ROBOT STUDIO
VIRTUAL MODEL OF PRODUCTION MACHINE IN ABB ROBOT STUDIO ENVIRONMENT

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Petr Běloušek**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Václav Kaczmarczyk, Ph.D.**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Petr Běloušek ***ID:*** 158105

Ročník: 2 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Virtuální model výrobního stroje realizovaný v prostředí ABB Robot Studio

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je použít ABB Robot studio na vytvoření simulace stroje firmy FLSmidth na zpracování vzorků sypkého materiálu a připojení na hotové řízení stroje napsané v prostředí Beckhoff TwinCAT2.

1. Seznamte se s prostředím ABB Robot Studio
2. Nastudujte princip funkčnosti předmětného stroje
3. Na základě podkladů dodaných firmou FLSmidth realizujte virtuální model
4. Proveďte konfiguraci směřující k propojení virtuálního stroje se skutečným řízením
5. Otestujte vaše řešení

DOPORUČENÁ LITERATURA:

[1] ABB Robot Studio technická dokumentace

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 1.6.2020

Vedoucí práce: Ing. Václav Kaczmarczyk, Ph.D.

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

Abstrakt

Diplomová práce se zabývá vytvořením virtuálního modelu v prostředí ABB RobotStudio simulující činnost stroje na zpracování sypkého materiálu od firmy FLSmidth. Stroj je řízen PLC Beckhoff CX8090. Komunikace mezi programy TwinCAT2 a ABB RobotStudio je řešena přes TCP/IP.

Klíčová slova

ABB RobotStudio, virtuální model, simulace stroje, TwinCAT2, PLC, CX8090, TCP/IP

Abstract

Diploma thesis deals with creating a virtual model of machine for processing bulk materials from FLSmidth in ABB RobotStudio environment. This machine is controlled by Beckhoff CX8090 PLC. Communication between TwinCAT2 controlling program and RobotStudio simulation is provided TCP/IP.

Keywords

ABB RobotStudio, virtual model, machine simulation, TwinCAT2, PLC, CX8090, TCP/IP

Bibliografická citace:

BĚLOUŠEK, P. *Virtuální model výrobního stroje realizovaný v prostředí ABB Robot Studio.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2019. 80 s. Vedoucí diplomové práce Ing. Václav Kaczmarczyk, Ph.D..

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Virtuální model výrobního stroje realizovaný v prostředí ABB Robot Studio jsem vypracoval samostatně pod vedením vedoucího diplomovou práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.“

V Brně dne: **11. května 2020** ……

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Václavu Kaczmarczykovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne: **11.května 2020** ……

podpis autora

Obsah

Seznam symbolů a zkratek

Zkratky:

ADS	…	Automation Device Specification
API	…	Application Programming Interface
CAD	…	Computer-aided design
CE	…	Compact Embedded
FIFO	…	First In First Out
GUI	…	Graphic User Interface
HMI	…	Human Machine Interface
HW	…	Hardware
IoT	…	Internet of Things
I/O	…	Input/Output
IRC5	…	Industrial Robot Controller 5
IP	…	Internet Protocol
MS	…	Microsoft
OS	…	Operační Systém
PLC	…	Programmable Logic Controller
POUs	…	Program Organization Units
RAM	…	Random Access Memory
RT	…	Real-Time
SCARA	…	Selective Compliant Articulated Robot Arm
SDK	…	Software Development Kit
ST	…	Strukturovaný Text
SW	…	Software
TCP	…	Transmission Control Protocol
UDP	…	User Datagram Protocol

Symboly:

A	…	ampér
MB	…	megabyte
MHz	…	megahertz

Seznam obrázků

1. ÚVOD

Tato diplomová práce se zaobírá simulací strojního zařízení od firmy FLSmidth, který slouží ke zpracování sypkého materiálu, v tomto případě cementu. Cílem je již existující strojní zařízení, které je ovládáno pomocí PLC značky Beckhoff, realizovat v simulačním prostředí ABB RobotStudio a graficky zobrazovat právě probíhající stavy.

V současnosti je jediným grafickým prvkem, informujícím o stavu zařízení, operátorský panel SIEMENS, kde je navíc pouze stavový diagram se stručným popisem. Pomocí programu RobotStudio by tak bylo možné detailně zobrazovat veškeré chování stroje včetně dynamických vlastností.

Samotná práce má tedy mimo vizuální prezentace stroje také zjistit, zda lze při případném vývoji nového zařízení využít tuto cestu k odladění chyb v řídícím softwaru i CAD modelu. Problematice virtualizace výrobních strojů je věnována první kapitola, ve které je čtenáři přiblížen tento čím dál rozšířenější průmyslový trend. Následně je představeno prostředí ABB RobotStudio se zaměřením na používané funkce. Poslední teoretickou částí je seznámení se s odesílací stanicí, která je předmětem simulace. Zde jsou popsány důležité části přístroje, včetně typů ovládání a výrobních cyklů.

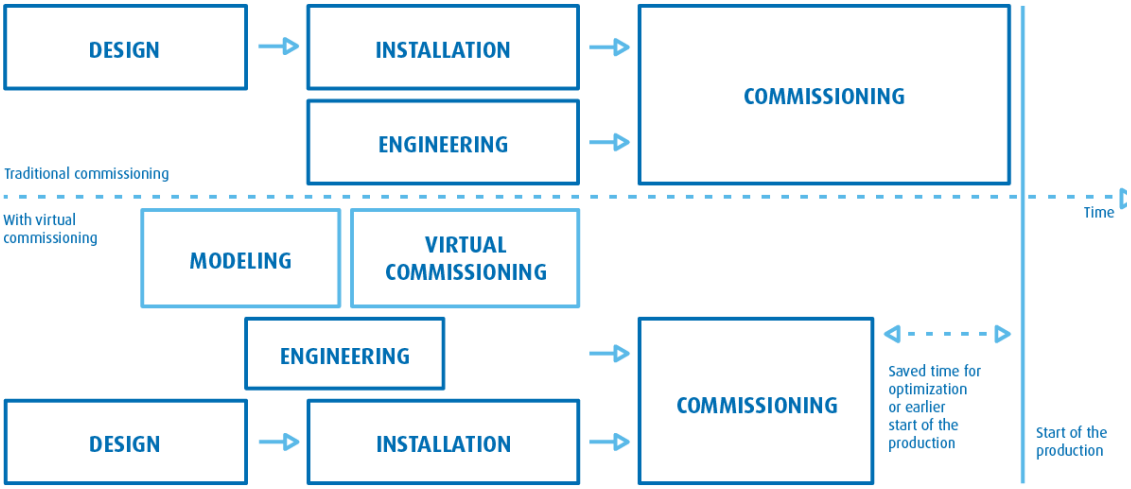
Poté již následuje praktická část, ve které je podrobně vysvětlen postup tvorby od importu modelu do RobotStudia, přes rozdělení do jednotlivých funkčních celků, přidělování fyzikálních vlastností, vytváření mechanizmů a práci se Smart Compomponenty. Po jeho dokončení přichází na řadu výběr způsobu komunikace mezi vytvořeným strojem v RobotStudiu a řídícím programem psaném v TwinCATu 2. Následně probíhá tvorba TCP/IP serveru na straně simulovaného stroje a k němu je postupně vytvářena i klientská strana reprezentující PLC.

Poslední kapitola se věnuje testování navrženého řešení, kde jsou zmíněny dosažené výsledky simulace včetně případných návrhů na vylepšení do budoucna.

2. VIRTUALIZACE VÝROBNÍHO STROJE

Je-li k řízení digitálního modelu využita reálná řídící jednotka (například PLC), která je následně použita k řízení vyvíjeného zařízení, dá se mluvit o virtuálním zprovoznění.

Právě tento trend virtualizace představuje účinnější provázání jednotlivých kroků vývoje od konceptu stroje (produktu) přes návrh konstrukce, strukturálně-mechanické simulace až po optimalizaci procesu výroby. Zároveň se jedná o jeden z pilířů, díky kterému lze až o 70 % zkrátit čas potřebný k uvedení skutečného stroje do provozu.[1][2][3]



Obrázek 1: porovnání průběhů obou postupů[4]

2.1 Role virtualizace ve výrobě

Výrobu v současné době ovlivňuje mnoho technologických změn vázaných na vývoj nových materiálů, rostoucí podíl aditivní výroby, automatizace a inteligence strojů. Změny výrobních technologií s sebou nesou požadavky na výrobní stroje, jejichž vývoj a parametry se musí měnit. Technická a finanční rizika spojená s těmito změnami je možno dílčím způsobem simulovat pomocí virtuálních modelů.[2]

2.2 Průběh virtuální zprovoznění

Základním prvkem pro virtuální zprovoznění stroje je 3D model doplněný o logické bloky, které definují chování jednotlivých komponent. K této vytvořené logice se na základě elektrotechnických a pneumatických schémat připojují signály, které vstupují buď do PLC nebo do stroje. Pro vyladění případných nedostatků před samotnou hardwarovou realizací je tedy nezbytné, aby se do simulace stroje zahrnulo co možná nejvíce vlastností, díky kterým se lze chováním maximálně přiblížit skutečnému stroji. [3]

2.3 Výhody použití

Velmi zmiňované jsou především výhody, které s sebou virtuální zprovoznění přináší.

- ověření a odladění řídícího programu včetně návrhu a napojení snímačů
- snadná optimalizace návrhu bez potřeby testování na reálném zařízení
- významná úspora času během inženýringu
- testování bezpečnostních prvků a blokací
- testování variant řídících programů a konstrukčních řešení
- odladění chybových scénářů
- zkrácení dobu nutné k oživování a ladění zařízením
- opravdový mechatronický přístup propojením mechaniky, elektroniky a softwaru[1]

3. ABB ROBOTSTUDIO

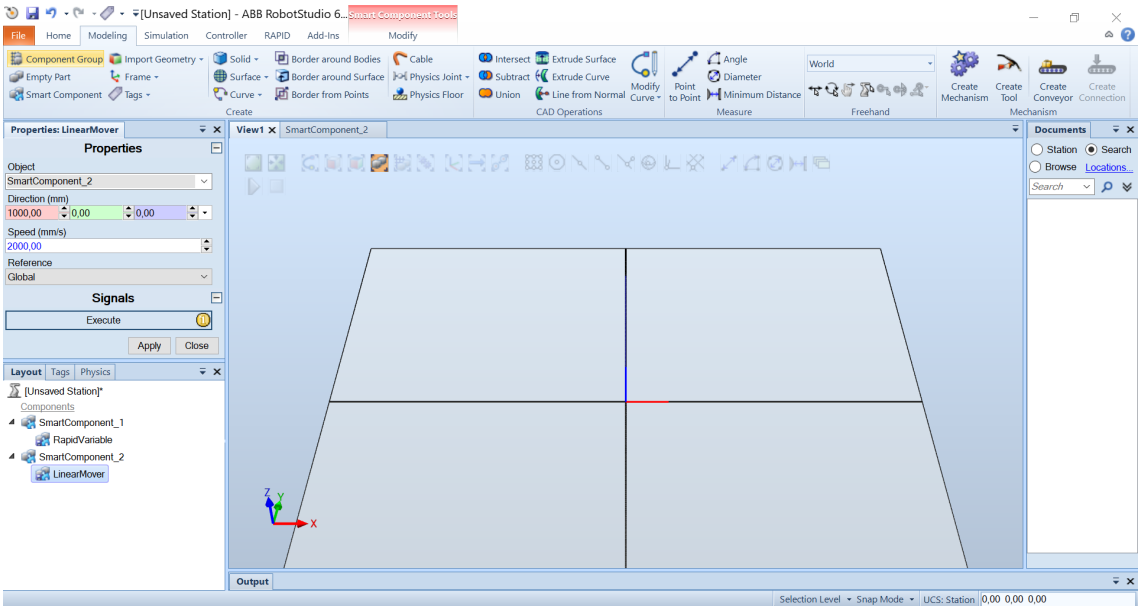
Právě RobotStudio bylo zvoleno jako simulační prostředí k realizaci předmětného stroje. Následující podkapitoly jsou tedy věnovány bližšímu seznámení se s prostředím a jeho funkcemi, které uživateli nabízí. Zaměřena je především na použité funkce, nicméně jsou zde zmíněny i další klíčové funkce pro práci s roboty.

3.1 Obecné informace

RobotStudio je inženýrský nástroj pro vytváření pracovišť s robotickými rameny ABB. Nejedná se tedy o univerzální software vhodný i pro jiné značky robotů. RobotStudio umožňuje nastavení a programování jak reálných robotů v online režimu, tak i jejich virtuálních kopií pomocí offline programování virtuálního kontroléru. Offline programování s sebou přináší především možnost rychlejší implementace výroby nových typů výrobků do již existující výrobní linky, předpřipravení programu, ověřování dosahu jednotlivých robotických ramen a simulace pracoviště do velkých detailů. Tyto faktory mají značný vliv na úsporu času a návratnost investic robotizovaných systému, na kterou se klade vysoký důraz. RobotStudio nabízí pokročilé možnosti pro modelování, simulaci, vizualizaci robotického pracoviště a celých výrobních buněk včetně bezpečnostních funkcí, 3D vidění a vzdálené správy robotu. [5]

3.2 Představení prostředí

Uživatelům MS Office 2007 bude na první pohled celý design jistě povědomý, neboť samotné rozložení záložek a rozmístění ovládacích prvků silně připomíná tento produkt. V horní části je celkem 7 hlavních záložek. V závislosti na aktivní otevřené záložce se mění i obsah levého okna.



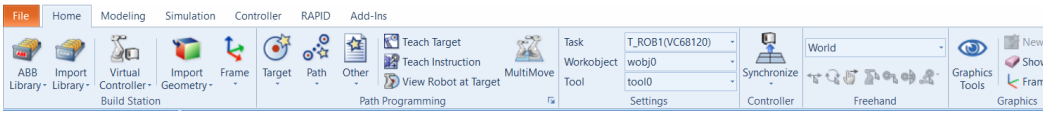
Obrázek 2: prostředí ABB RobotStudio

3.2.1 Záložka File

Jedná se o výchozí stránku zobrazující se při spuštění RobotStudia. Nachází se zde obligátní položky jako je ukládání, načítání, vytváření nových projektů či pouze specifické části programu, informace o stávajícím projektu a další. Za zmínku stojí podzáložka *Share.* Ta nabízí funkce zabalení kompletního řešení projektu včetně používaného kontroléru do 1 souboru, uložení stanice do spustitelného formátu, kde si i uživatel bez nainstalovaného RobotStudia může s omezenou funkcionalitou prohlížet navrhované řešení, spouštět simulaci a to i s doplňkem pro virtuální realitu. Dále je zde přístup ke komunitou vytvořeným knihovnám, doplňkům a v poslední řadě možnost pořádání online meetingů, kde lze snadno prezentovat svá řešení ostatním i s podporou virtuální reality.

Poslední užitečnou podzáložkou je *Help*, kde na 1 místě jsou odkazy na stránky ABB, uživatelského fóra, video návody, podporu, vývojářské centrum s ukázkovými příklady a veškeré online i offline dostupné dokumentace k RobotStudio API, IRC5, Add-ins a další.

3.2.2 Záložka Home



Obrázek 3: nabídka funkcí v záložce Home

3.2.2.1 Blok Build Station

První položkou zleva jsou knihovny přímo od výrobce ABB. Zde uživatel najde téměř všechny roboty nabízené touto společností. Jsou rozřazeny do kategorií kloubových robotů, kolaborativních robotů, paralelních robotů, SCARA robotů, lakovacích robotů, polohovadel a kolejových drah.

V *Import Library* jsou obecnější knihovny, kde lze nalézt uživatelem použité a vytvořené knihovny, stejně tak tu jsou připraveny základní díly pro sestavování robotů a výrobních linek jako jsou podstavce, pracovní nástroje, kamery, Flex Pendant, dopravníky, zábrany, boxy na kontrolér IRC5 a další.

Ve virtuálních kontrolérech jsou možnosti vložení nového dle šablony, vytvoření zcela nového a použití existujícího modelu. Při vytváření nových typů je zde pouze pár obecných nastavitelných parametrů. Pokud má uživatel specifičtější požadavky, je doporučené kontrolér vytvářet a upravovat až v záložce *Controller*, kde jsou širší možnosti úprav.

Import Geometry slouží především k načtení modelu z jiných CAD programů. Je zde široká kompatibilita formátů, kde jsou podporovány soubory typu RSGFX, VRML, ACIS, STL, COLLADA, OBJ, 3DS a LDraw.

3.2.2.2 Blok Path Programming

Přes tlačítko *Target* je možné definovat nové cílové body, pro vytvoření nové trajektorie robotického ramene se užívá tlačítko *Path*. Při volbě *Autopath* lze snadno pomocí několika kliknutí myši vytvořit trasu podél celého obrobku. V dalších možnostech je i vytvoření dodatečných vlastností pro nástroj, jako je dovolené zatížení a polohování.

Druhá část bloku je především pro naučení těchto koncových bodů a instrukcí včetně možnost i ovládání více robotů naráz.

3.2.2.3 Blok Setting

Blok slouží pouze pro výběr již existující úlohy, workobjectu a a nástroje robotu.

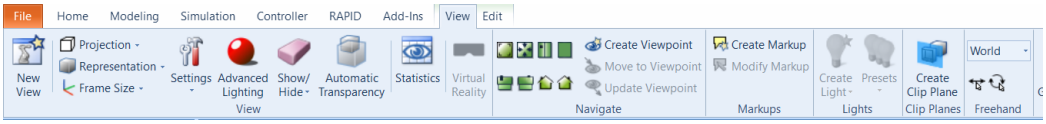
3.2.2.4 Blok Controller

Slouží pro synchronizaci stanice s programem RAPID. Volbou *Synchronize to RAPID* je možné graficky nadefinované polohy a trajektorie převést do psaného kódu. Pro opačný případ slouží volba *Synchronize to Station.*

3.2.2.5 Blok Freehand

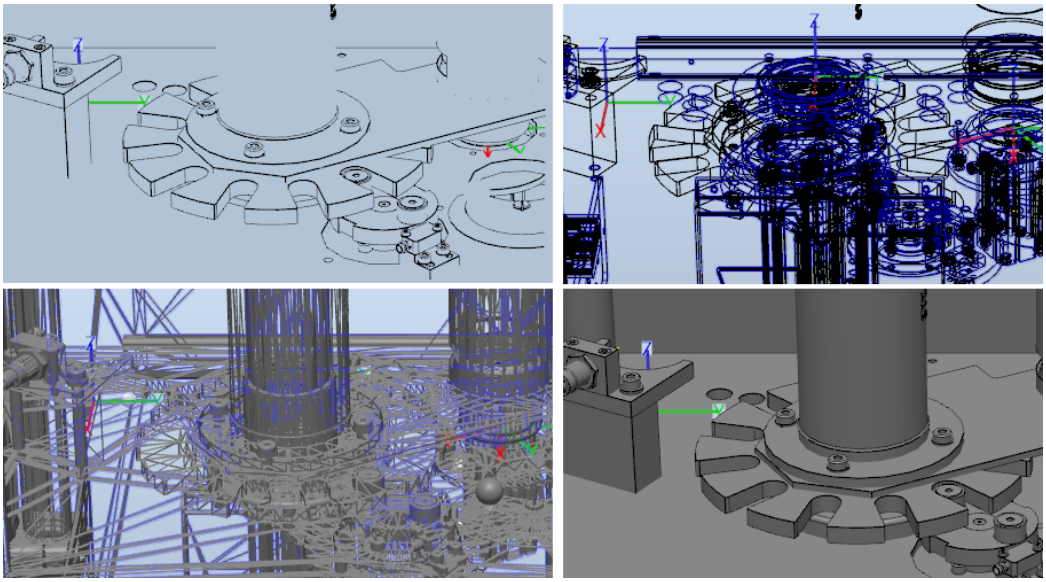
Umožňuje manipulaci s objekty a robotem. První 2 položky slouží pouze pro běžné změny poloh objektů, 3. je pro objekty se zapnutými dynamickými vlastnostmi. Zbylé prvky se už soustředí na manipulaci s robotem a jeho nástroji ve všech osách s lineárním i rotačním pohybem.

3.2.2.6 Blok Graphics



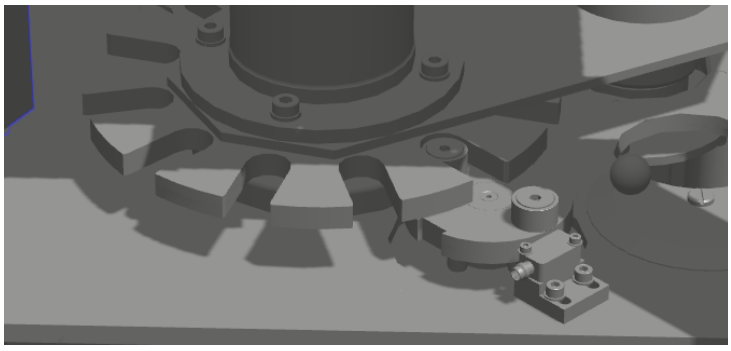
Obrázek 4: nabídka funkcí v záložce Home-Graphics

Po rozkliknutí prvku *Graphic Tools* se zobrazí nové podrobnější menu se záložkami *View* a *Edit*. První zmíněná slouží především pro úpravu zobrazení, kde pomoci *New View* lze přidat další scénu. Užitečným nástrojem *Representation*, kde jsou krom standardního *Surface* zobrazení ještě další 4 možnosti. (Hidden line removal, Wireframe, Wireframe triangles a Surface+Wireframe)



Obrázek 5: příklady zobrazení při užití různých reprezentací

Při použití *Advanced Lighting* se zobrazí stíny a aktivuje volba *Create Light*, kde je možné si nastavit 3 druhy osvětlení s různými vlastnosti vrhání stínu, intenzitou, pozicí a dalšími.



Obrázek 6: použití světelných zdroju včetně umístění

Poslední dvě funkce, které jsou v této sekci zmíněny, jsou *Statistics*, kde je kompletní výčet prvků, hran a dalších, které náleží jednotlivým dílům a mechanizmům,

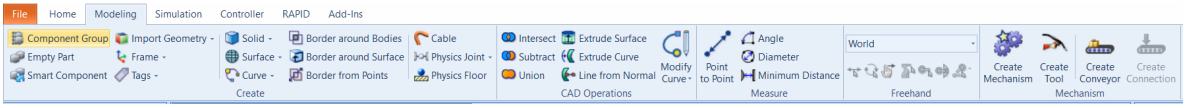
Part Name	Bodies	Faces	Triangles	Vertices
mechanismGear/L1	1000 (45,0%)	19861 (46,4%)	456578 (41,2%)	433723 (41,8
mechanismGear/L2	45 (2,0%)	647 (1,5%)	35476 (3,2%)	33307 (3,2%
mechanismGear/L3	26 (1,2%)	281 (0,7%)	9880 (0,9%)	9294 (0,9%)
mechanismGripper/L1	18 (0,8%)	203 (0,5%)	6336 (0,6%)	5854 (0,6%)
mechanismGripper/L2	11 (0,5%)	182 (0,4%)	6654 (0,6%)	5835 (0,6%)

Obrázek 7: statistický přehled součástí a jejich propracovanosti

a *Automatic Transparency,* který po přiblížení k modelu částečně zneviditelní většinu částí a nechá plně viditelné pouze požadované díly, roboty a mechanizmy.

Záložka *Edit* je již čistě estetická a slouží především pro přesnější barevné zobrazení. K dispozici jsou opět předdefinované barevné struktury pro různé kovy, skla, barvy a další.

3.2.3 Záložka Modeling



Obrázek 8: nabídka funkcí v záložce Modeling

3.2.3.1 Blok Create

Pro vytvoření a import okolních předmětů slouží právě tento blok. *Empty part* má využití především pro sdružení několika CAD prvků (Body) do jednoho celku, se kterým poté lze i takto pracovat. *Component Group* slouží obdobně, tentokrát však shlukuje celé části do jedné skupiny.

Velice důležitou a užitečnou funkcí je *Smart Component.* Pomocí této funkce lze vytvářet objekty a simulovat procesy okolního prostředí robotického ramena. Jsou zde základní stavební prvky, kterým lze přidělovat určité vlastnosti a spojitosti s ostatními komponenty, a vytvořit tak mnohem komplexnější prvek definovaný jako Smart Component. Uživatel má na výběr z několika kategorií s předpřipravenými prvky.

- Signals and Properties – zpracování I/O signálů včetně matematických výpočtů, logických funkcí, časovačů,...
- Parametric Primitives – vytváření parametrických objektů a uskupeních
- Sensors – snímače kolizí, přítomnosti, pozice, rozměrů,...
- Actions – akce s předměty jako je zobrazení, zmizení, uchopení,...
- Manipulators – pro pohyby objektu po libovolně definované křivce
- Others – jedná se především o informační a audiovizuální komponenty pro operátora, ale obsahuje i prvky pro inicializaci komponent

V případě potřeby ABB nabízí knihovny, s pomocí kterých si lze v programovacím prostředí vytvořit vlastní Smart Component s požadovanými vlastnosti, vstupy

a výstupy. Toto řešení je následně možné naimportovat do RobotStudia jako běžnou knihovnu.

Další nabízené funkce v tomto bloku jsou například vytváření kabelů robotu, podlahy, která je nutná při simulaci s dynamickým chováním, a *Physics Joint*, přes který si lze nadefinovat různé druhý spojení dvou částí k sobě jako jsou například panty, či jiná kloubová spojení.

3.2.3.2 Blok CAD Operations

Slouží pro základní práci s vytvořenými CAD objekty ze záložky *Create*.

3.2.3.3 Blok Measure

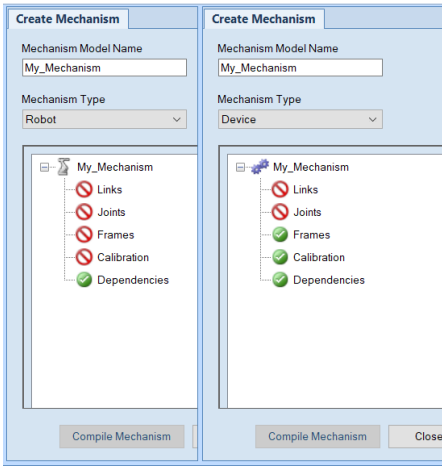
Nabízí užitečné funkce pro měření vzdálenosti, či úhlu mezi 2 body.

3.2.3.4 Blok Mechanism

Při požadavku na pohybování více částí naráz závisle či nezávisle na sobě, je vhodné použít nástroj *Create Mechanism*, který umožňuje manipulovat s těmito předměty po uživatelem nadefinovaných polohách.

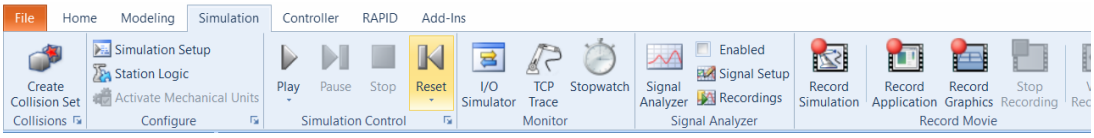
Krom pole pro pojmenování vytvořeného mechanizmu je zde lišta s výběrem typu mechanizmu s celkem čtyřmi možnostmi, kde každá má své specifické vlastnosti a požadavky, které musí uživatel při vytváření dodržovat viz obrázek číslo 9. Červené ikonky značí, co vše je pro daný typ potřeba dodefinovat.

- Robot – Mechanizmus s funkcí TCP Trace, pohyby mohou být řízeny přes virtuální kontrolér.
- Tool - Opět podporuje funkci TCP Trace, pohyby mohou být řízeny robotem.
- External axis – Zde již není TCP Trace, mechanismus však může být ovládán přes virtuální kontrolér. Využití je tedy například pro polohovače obrobku, portálové roboty a další, na které lze následně připevnit roboty.
- Device – Není zde podporována funkce TCP Trace, stejně tak nelze tento mechanizmus řídit přímo virtuálním kontrolérem. Používá se tedy především pro nadefinování mechanizmu u obecný částí, které obvykle nemají nic společného s robotickými rameny. [6]



Obrázek 9: ukázka požadavků při tvorbě různých mechanizmů

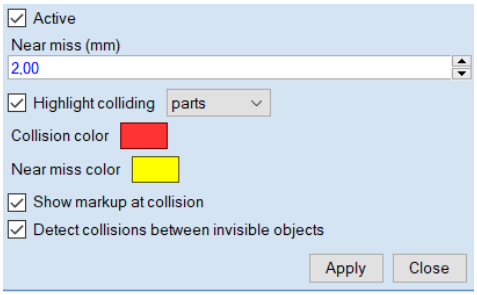
3.2.4 Záložka Simulation



Obrázek 10: nabídka funkcí v záložce Simulation

3.2.4.1 Blok Collisions

Zde je možné zjišťovat případně kolize simulace. Zvolí se 2 objekty, mezi kterými k ní může docházet. Jakmile nastane kolize, ve výstupním okně a daný objekt se zabarví. Pro určitou toleranci lze nastavit pole *Near miss (mm)*.



Obrázek 11: nastavení Collision set

3.2.4.2 Blok Configure

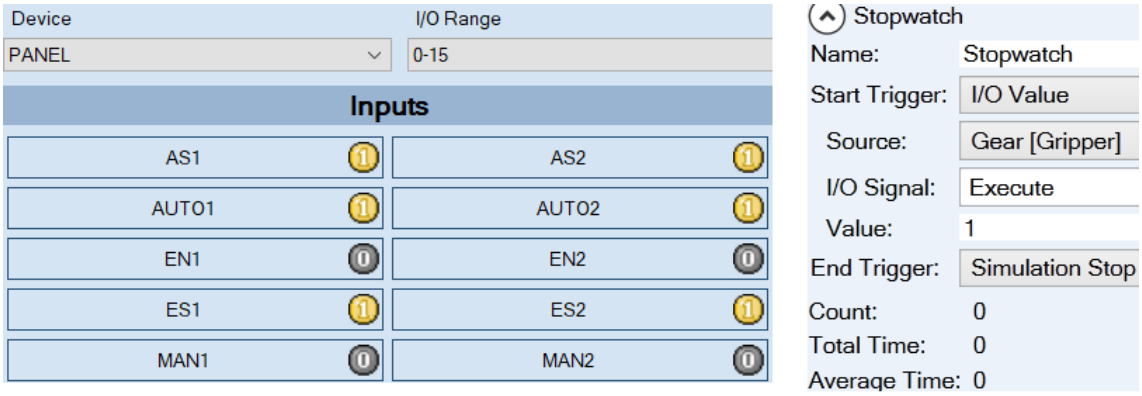
Položka *Station Logic* se používá především pro signálové propojení kontroléru s vytvořeným Smart Componentem. Menu i nabídka je stejná jako při tvorbě samotného Smart Componentu. *Simulation setup* slouží pro předpřipravení stavů a scén během simulace. Další skrytou funkcí, dnes již méně používanou, je *events*, ve které si uživatel nastaví událost, která následně spustí definovanou akci.

3.2.4.3 Blok Simulation Control

Tyto 4 tlačítka slouží pouze pro přehrávání simulace. Při spuštění je zde také možnost zároveň spustit i RAPID kód, případně celou vizualizaci nahrávat.

3.2.4.4 Blok Monitor

Pomocí funkce *I/O Simulator* lze monitorovat a měnit hodnoty vstupů a výstupů virtuálního kontroléru a Smart Componentu, což je užitečné především během fáze testování napojení. *TCP Trace* je pro zobrazení celé trajektorie, kterou robot vykonává. Dále je zde *Stopwatch* sloužící pro měření času mezi libovolně nastavitelnými událostmi v simulaci.

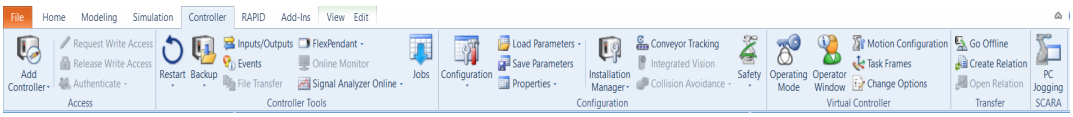


Obrázek 12: ukázka funkcí I/O Simulator a Stopwatch

3.2.4.5 Blok Signal Analyzer

Slouží pro trasování hodnot určitých signálů v čase. Po vybrání požadovaných signálů na sledování v *Signal Setup* stačí jen pustit simulaci a v *Signal Analyzer* se již zobrazují vybrané průběhy.

3.2.5 Záložka Controller



Obrázek 13: nabídka funkcí v záložce Controller

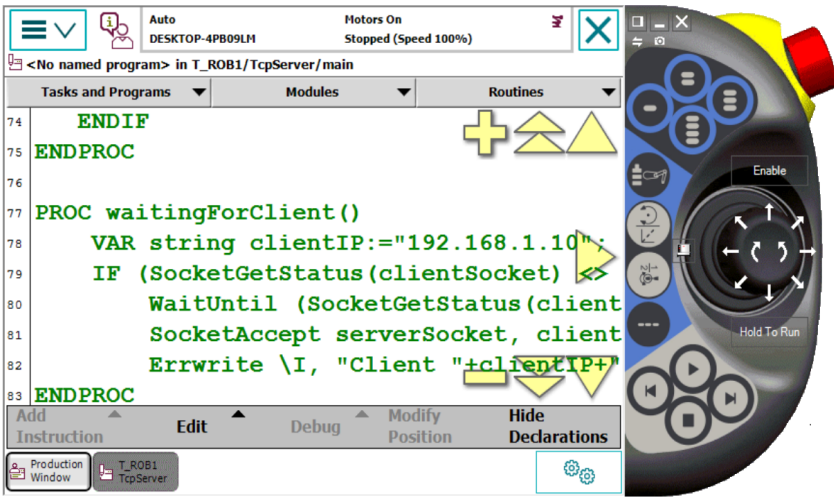
3.2.5.1 Access

Přes *Add Controller* lze přidat reálně připojený či virtuální kontrolér. Pro rozdílná práva přístupu slouží zbylé položky, kterými lze žádat o povolení k zápisu, či se přes ně přihlásit jako jiný uživatel a kontrolovat nastavení přístupu.

3.2.5.2 Controller Tools

První položka *Restart* nabízí několik možností restartu kontroléru včetně úplného vypnutí. Po každé provedené úpravě, například signálů, je pro zobrazení nezbytné restartovat kontrolér přes *Warmstart*. Práce se zálohami kontroléru probíhá přes tlačítko *Backup.* V *Inputs/Outputs* jsou zobrazeny všechny uživatelem dodefinované signály s možností měnit jejich hodnoty. Pro zobrazení veškeré historie událostí slouží *Events.*

Je zde také možnost využít napojení na *FlexPendant* a to buď reálný či virtuální, ke kterému je za použití SDK možnost vytvořit vlastní aplikaci.



Obrázek 14: použití funkce Virtual Flex Pendant

Dále jsou zde funkce na online monitorování robotu a *Jobs* na plánování práce s vlastním záložkou.

3.2.5.3 Configuration

Důležitým prvkem je stejnojmenné *Configuration,* které slouží k zobrazení a nastavení všech důležitých parametrů z oblasti komunikace, kontroléru, I/O systému, pohonů a dalších. Dále jsou zde prvky pro detailní informace o připojeném reálném kontroléru, vytváření nových a funkce pro pokročilé nastavení dopravníkových pásů a počítačového vidění, ke které mají svá vlastní okna s novými funkcemi.

Pro robotická ramena je tu také *Collision Avoidance* k předejití srážky mezi ramenem a okolními objekty. Přes *Safety* lze zobrazovat a měnit bezpečné zóny robota při práci.

3.2.5.4 Virtual Controller

Operating mode je určen k natavení režimu ovládání daného kontroléru. Na výběr je režim automatický, manuální a manuální s plnou rychlostí včetně tlačítek na spínání motorů a nouzové zastavení. Dále jsou tu funkce pro úpravu a nastavení dodatečných os spojených s robotem a *Change Option* pro případně změny nastavení ve vytvořeném virtuálním kontroléru.

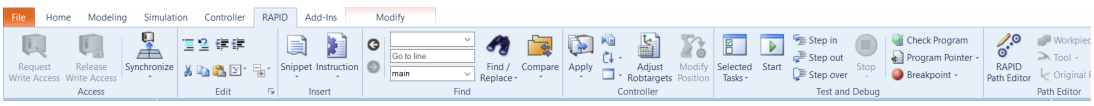
3.2.5.5 Transfer

Pomocí *Go Offline* je možné vytvořit virtuální kopii daného kontroléru a přes *Create Relation* jde snadno přenášet vybraná data mezi zvolenými kontroléry. Funkcí *Change Option* se provádí případné dodatečné změny v nastavení kontroléru.

3.2.5.6 SCARA

PC Jogging lze použít pro SCARA roboty, kde je další okno s prvky pro manipulaci, učení a kalibraci.

3.2.6 Záložka RAPID



Obrázek 15: nabídka funkcí v záložce RAPID

3.2.6.1 Blok Access

Obsahuje již použité funkce z předešlých záložek.

3.2.6.2 Blok Edit

Editační blok se vztahuje k otevřenému RAPID kódu. Uživatel zde nalezne základní funkce pro práci s textovým programem, jako je zakomentování a odkomentování části kódu, vkládání, kopírování, úrovňové formátování a skrývání jednotlivých programových bloků podle *regionů*.

3.2.6.3 Blok Insert

Rozkliknutím tlačítka *Instruction* si uživatel může zobrazit kompletní seznam veškerých dostupných příkazů v jazyku RAPID, který je roztříděn do základních kategorií užití. Tlačítko *Snippets* slouží spíše pro vkládání deklarací nástrojů, matic, procedur a dalších.

3.2.6.4 Find

Pro běžnou práci s vyhledáváním textu, případně pro porovnání verzí programu mezi editorem a kontrolérem.

3.2.6.5 Controller

Pomocí *Apply* se uloží změny ve všech modulech. Tlačítko *RAPID Tasks* slouží pro zobrazení detailních informací o všech aktivních úlohách v kontroléru.

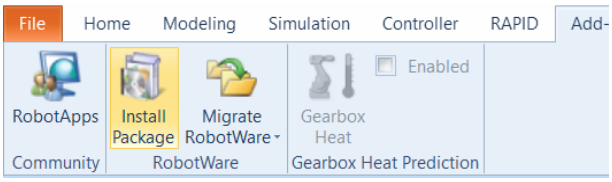
Dále je zde možnost výběru provádění programu mezi pouze jedním průchodem a opakovaně se vykonávajícím.

3.2.6.6 Test and Debug

RobotStudio nabízí i běžné funkce pro ladění programu jako jsou breakpointy, krokování a kontrola kódu. Při práci s více úlohami lze mezi nimi přepínat pomocí *Selected Task*.

Program pointer zobrazuje aktuální prováděný řádek v kódu. Pokud se ukazatel zastaví v určité proceduře, je možné ho resetovat přes *Set program pointer to main in all tasks.*

3.2.7 Záložka Add-Ins



Obrázek 16: nabídka funkcí v záložce Add-Ins

3.2.7.1 Blok Community

Zde má uživatel přístup k doplňkům od ABB vývojářů i od samotných uživatelů. Lze si zde stáhnout různé verze RobotWaru, doplňků pro RobotWare, stejně tak obecné doplňky pro RobotStudio. Jsou zde i nejpopulárnější uživateli vytvořené či naprogramované Smart Componenty, které si lze stáhnout do svého projektu.

3.2.7.2 Blok RobotWare

Tyto položky slouží k ručnímu doinstalování RobotWaru z již existujících souborů, vytvoření nového kontroléru se specifickými funkcemi a verzemi RobotWaru a také k aktualizaci verze obsažené v reálném robotu.

3.2.7.3 Blok Gearbox Heat Prediction

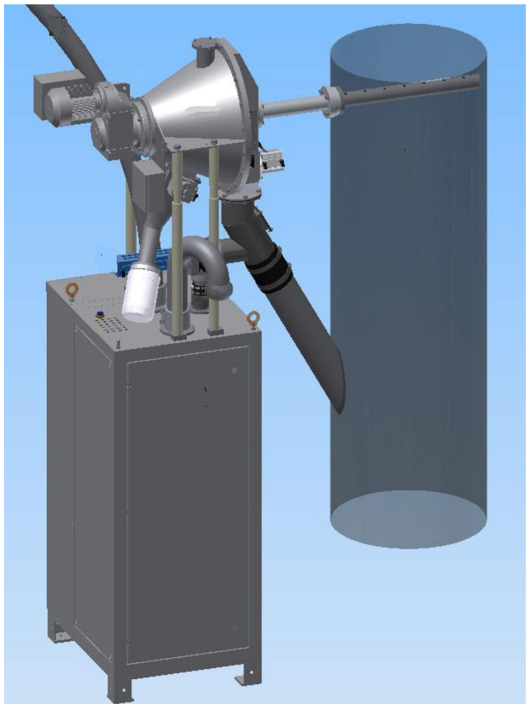
Jedná se o funkci, pomocí které lze simulovat chování robotu z hlediska přehřívání jeho jednotlivých částí. Simulace bere v potaz okolní teplotu počet pracovních cyklů, pauzy mezi nimi, zda jsou přítomny chladiče a další. Díky tomu by měl uživatel dostat informace, do jak míry mohou být zatěžovány motory a převodovky.

4. STROJ PTS102

Jedná se o strojní zařízení od firmy FLSmidth, které je právě předmětem realizace a následné simulace v prostředí ABB RobotStudia.

4.1 Funkce přístroje

Zařízení funguje jako automatická odesílací stanice vzorků sypkého nehořlavého materiálu, například cementu, který je přepravován v patronách. Je součástí systému, ve kterém je pomocí vzorkovače odebráno určité množství daného materiálu do patrony. Ta je následně přes potrubní poštu odeslána stanicí do centrální laboratoře.



Obrázek 17: propojení stanice[7]

4.2 Popis částí stroje

Následující podkapitoly seznamují s jednotlivými částmi odesílací stanice. Jde především o vytvoření představy, co vše by mohlo a mělo být simulováno.

4.2.1 Řídící skříň

Tato část obsahuje elektrická zařízení jako je hlavní zdroj napájení, pojistky, další bezpečnostní komponenty a také samotný řídící systém realizovaný programovatelným logickým kontrolérem(PLC).[7]

4.2.2 Dotykový ovládací panel

Panel je umístěn na přední straně stanice. Vizualizace slouží k řízení provozu celého strojního zařízení. Dále obsahuje všechny důležité parametry a stavy v textové či grafické formě, včetně chybových hlášení a varování. Řídící systém má stálé datové spojení s centrálním laboratorním systémem.

4.2.3 Motorem řízené rameno

Elektromotor, který přes ozubené kolo ovládá pozici ramene, zde slouží k přepravě patrony se vzorkem mezi jednotlivými stanicemi:

- plnící pozice, která je zároveň také výchozí pozicí
- pozice otvírače
- transportní pozice
- pozice pro případné ruční vyjmutí patrony z oběhu

4.2.4 Píst na plnící pozici

Pneumatický plnící píst slouží k vyzvednutí a přitisknutí patrony bez víčka k plnící trubici, na kterou je připojen vzorkovač. Těsnění zajišťuje bezprašnost během plnění. Po naplnění je píst opět sesunut do spodní pozice.

4.2.5 Otvírač

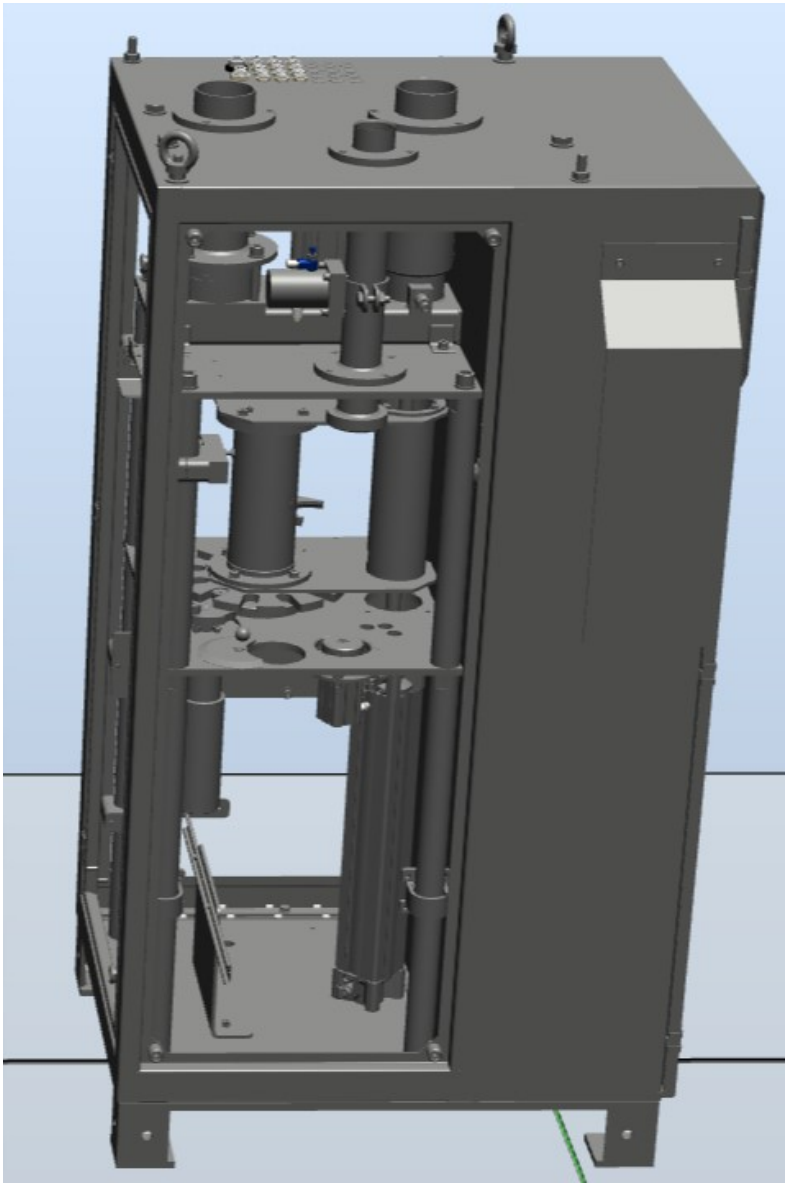
Jakmile přijede rameno na pozici otvírače, víčko z patrony je automaticky odtrženo a vyzdviženo do horní pozice otvírače. V ní zůstává po celou dobu plnění patrony. Po dokončení plnění se patrona se vzorkem vrací na toto místo, kde je následně víčko zatlačeno zpět do patrony.

4.2.6 Transportní píst

Slouží k přesunu patrony z manipulační trubice ramene do transportní trubice, odkud je potrubní poštou odeslána do laboratoře.

4.2.7 Stanice na výměnu patrony

Po najetí ramenem na tuto pozici a po odstranění mechanické zábrany na dolní části pracovní oblasti je možné patronu ručně vyjmout z oběhu. [7]



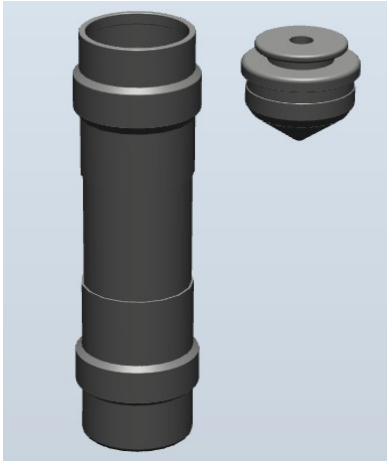
Obrázek 18: PTS 102, pro názornost bez bočního krytí

4.2.8 Dálkové krokové řízení

Jedná se o alternativní způsob řízení k hlavnímu automatickému režimu. Ovladač umožňuje rozdělení celého procesu na jednotlivé kroky, ve kterých následně stanice postupuje. Po každém dokončeném kroku se čeká na další povel od obsluhy.

4.2.9 Pneumatický systém

Stanice obsahuje i samostatné jednotky, které slouží pro regulaci jednotlivých pneumatických akčních členů. Jedná se například o pohon všech pístů sloužících pro manipulaci s patronou a kontrolu správnosti tlaku při odesílání.[7]



Obrázek 19: patrona s víčkem

4.3 Režimy ovládání

- Manual – ruční

  Tento režim slouží pouze pro spuštění, kontrolu, nastavení a údržbu stroje. Nelze v něm také spouštět automatické funkce stanice.
- Single Step – poloautomatický

  Jde o jednokrokový provozní režim, který lze řídit pouze přes připojené dálkové ovládání.
- Semi – automatický (lokální)

  Při této volbě se na zařízení automaticky provádí celý proces poté, co byla

snímačem zjištěna přítomnost patrony na vstupu. Tento režim vyžaduje zásah operátora, který musí ručně vyjmout naplněnou patronu se vzorkem a to v pozici na výměnu patron.

- Remote – automatický (vzdálený)
  Posledním způsobem je automatické vzdálené ovládání, kde celý proces probíhá sám a přítomnost operátora již není vyžadována. Operace přijmutí patrony, zpracování, naplnění a odeslání zpět do vzdálené laboratoře již probíhá automaticky.[7]

4.4 Popis procesu zpracování

Před začátkem každého cyklu odesílací stanice musí být splněny následující podmínky:

- Dvířka a panely jsou správně uzavřeny.
- Píst pro plnění je v horní poloze, patrona bez víčka je dostatečně přitlačena k těsnění u plnící trubice.
- Otvírač patrony je v horní pozici a svírá víčko.
- Píst pro transport patrony je ve spodní pozici.
- Poloha otočného ramena odpovídá pozici motoru.

Poté již může začít samotný proces. Vzorkovač pomocí dávkovací jednotky upevněné na odesílací stanici naplní patronu požadovaným množství sypké látky. Jakmile je plnění dokončeno, píst s patronou sjede do základní úrovně a rameno přesune patronu do pozice, kde se nachází otvírač s víčkem. Ten s dostatečnou silou víčko přitlačí k patroně, a utěsní tak celý obsah.

Následně je patrona ramenem přesunuta do pozice pro transport, ve které je za pomoci dalšího pístu vyzvednuta do transportního potrubí, odkud se již pod tlakem odešle přes potrubní poštu do laboratoře. V této pozici stanice setrvá do příchodu nové čisté patrony, o které informuje snímač umístěný v horní části.

Nyní je patrona pomocí pístu přepravena zpět k ramenu, kterým je přesunuta k úchytnému systému otvírače. Víčko je otvíračem odtrženo a vyzdviženo do horní polohy. Rameno poté dopraví otevřenou patronu na pozici plnění, kde je pístem vyzdvižena a vzduchotěsně přitlačena k plnící trubici. Tím se cyklus uzavírá a je možné celou operaci opakovat.[7]

5. REALIZACE VIRTUÁLNÍHO MODELU

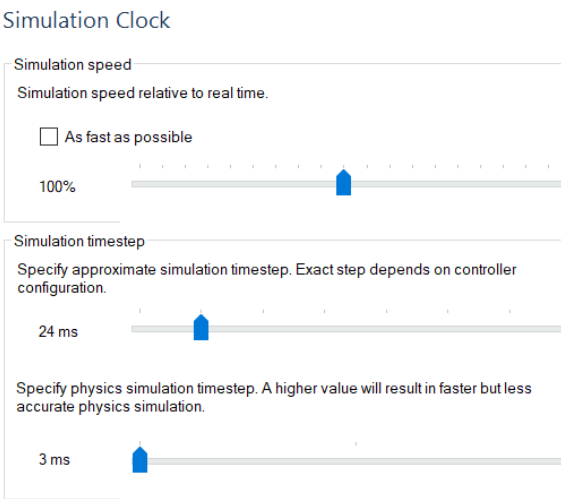
Kapitola popisuje autorův postup tvorby simulovaného stroje od prvotního nastavení prostředí přes rozdělení do jednotlivých funkčních celků včetně přidělení požadovaných vlastností tak, aby se celý stroj navenek jevil pouze jako jeden Smart Component se vstupy a výstupy. V průběhu jsou diskutována i alternativní řešení.

5.1 Import modelu do prostředí RobotStudia

Výchozím bodem pro tvorbu modelu odesílací stanice PTS102 byl export z programu Inventor ve formátu *.sat*, který je RobotStudiem podporován. Při tvorbě exportu byla zvolena nejvyšší úroveň detailů a geometrie. Totéž je třeba dodržet i při importu do RobotStudia.

Zde je třeba přenastavit výchozí hodnoty, neboť v základu je vybrána pouze střední kvalita. V *Options* je poslední záložka *Simulation ->Physics,* kde se nastaví ukazatel na *More accurate.*

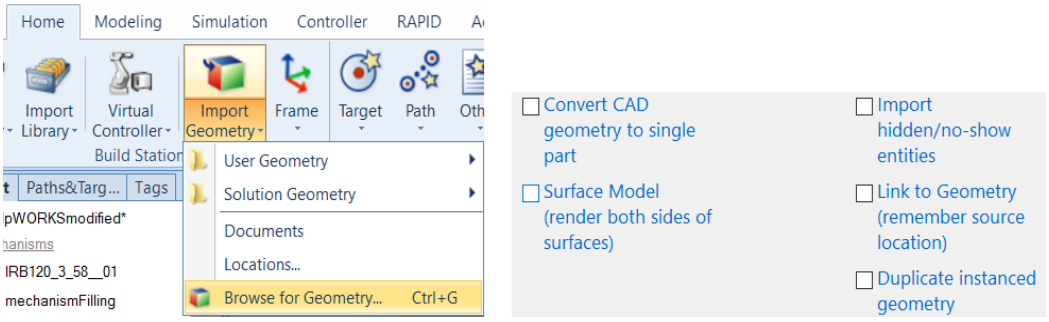
Dalším nastavení ovlivňující kvalitu kolizí je v *Simulation→Simulation Clock*, kde je třeba přenastavit především třetí parametr na co nejnižší hodnotu, v tomto případě na hodnotu 3 ms. Ta by měla zajistit menší časové úseky během simulace a s tím spojenou vyšší přesnost. Pokud by vlivem náročné simulace nestíhala grafická karta plynule zobrazovat, lze snížit rychlost simulace na hodnotu nižší než 100 % viz první ukazatel.



Obrázek 20: nastavení simulace

Posledním parametrem, který je nutno před importem modelu změnit, je úroveň detailů geometrie v *Graphics →Geometry*. Zde je možnost zaškrtnout více možností naráz, nicméně při snaze o nejvyšší kvalitu má smysl ponechat pouze možnost *Fine*.

Nyní již lze založit nový projekt na úvodní stránce. Po rozkliknutí záložky *New* se zobrazí následující nabídka, kde pro tento účel jsou zajímavé možnosti *Solution with Empty Station* a *Solution with Station and Virtual Controller*. Zde je rozdíl, že u druhého zmíněného si již na začátku může vybrat robot, se který se bude pracovat, včetně verze RobotWare. Nicméně to si lze dodefinovat i kdykoliv v průběhu vytváření. Přestože se u modelu odesílací stanice nebude nacházet žádné robotické rameno, tak samotný kontrolér, jež je součástí robotu, lze využít pro komunikaci s PLC. Po založení projektu se vybere importovaný *.sat* model přes *Import Geometry→Browse for Geometry.*



Obrázek 21: import modelu do RobotStudia

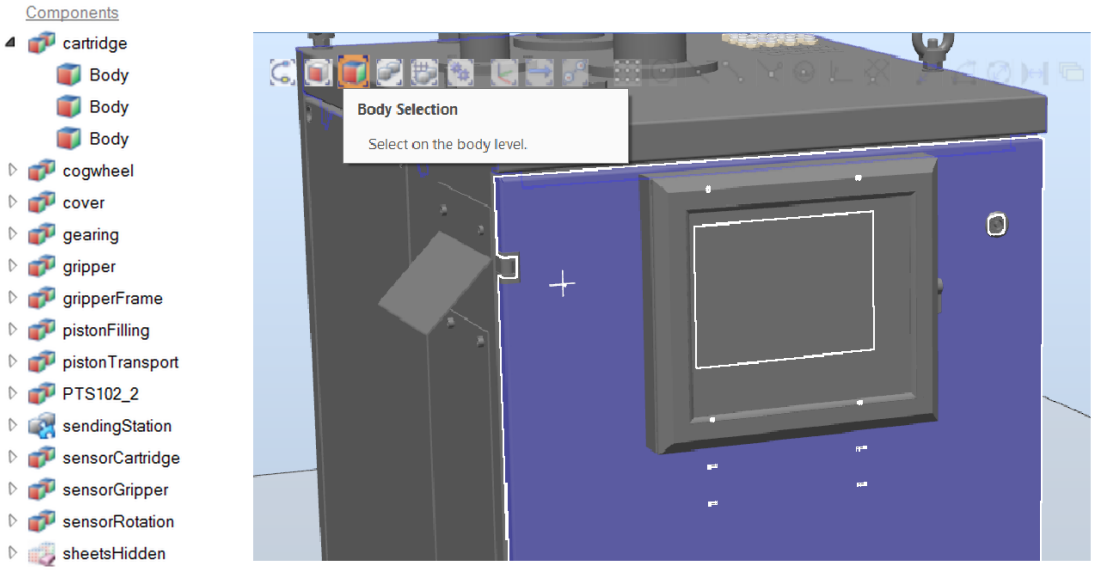
V otevřeném okně pro výběr modelu jsou k dispozici ještě nastavitelné parametry viz obrázek číslo 21, kde musí zůstat nezaškrtnuté políčko *Convert CAD geometry into single part*. Při zaškrtnutí by nebylo možné celý model rozdělit na samostatně se pohybující části.

5.2 Rozdělení modelu do funkčních celků

Po načtení dat je mimo samotného modelu na levé straně v záložce *Layout* vidět i část PTS102. Vnořené položky s názvem *Body* jsou veškeré součásti či pouze plochy součástí, které se v modelu nachází. Přibližně jich je kolem devíti set. Aby bylo možné manipulovat s jednotlivými funkčními celky, jako je motor, rameno, písty a další, je

nezbytné je přerozdělit do nově vytvořených dílů pomocí ikony s názvem *Empty part*, do kterých se budou postupně přesunovat prvky *Body*, které spolu souvisí.

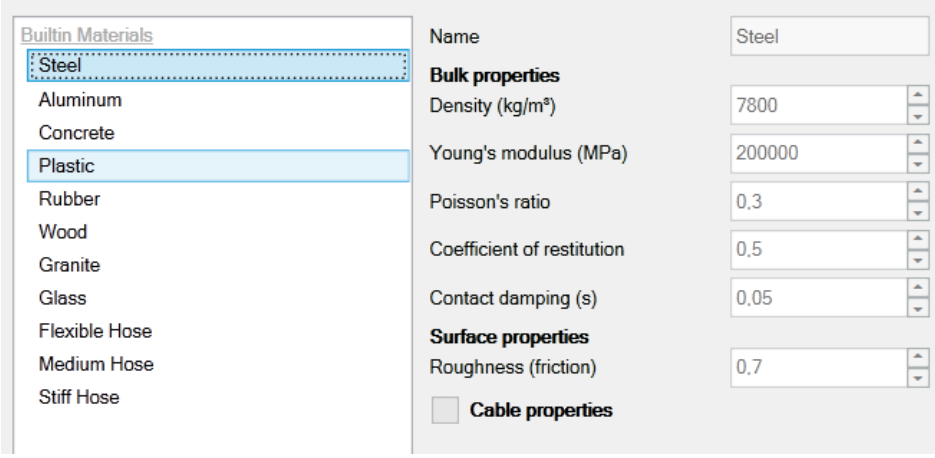
Z důvodu velkého množství prvků je nejsnazší cesta rozřazení přes využití mnoha funkcí v horní části zobrazení, kterými lze snadno označovat například jednotlivé díly stroje, jako je tomu v tomto případě. Při výběru se v levé liště zvýrazní právě označené prvky *Body*, které lze nyní myší lehce přetáhnout do nově vytvořené části *Part*. Takto rozřadit je třeba pouze součásti, se kterými se plánuje samostatně pohybovat. Zbylé díly mohou být ponechány v původní struktuře. Rozložení součástí do jednotlivých celků je možno měnit do okamžiku, než je daná část použita v nějakém mechanizmu, čemuž se věnuje kapitola 5.4.



Obrázek 22: rozřazování prvků za pomoci Body Selection

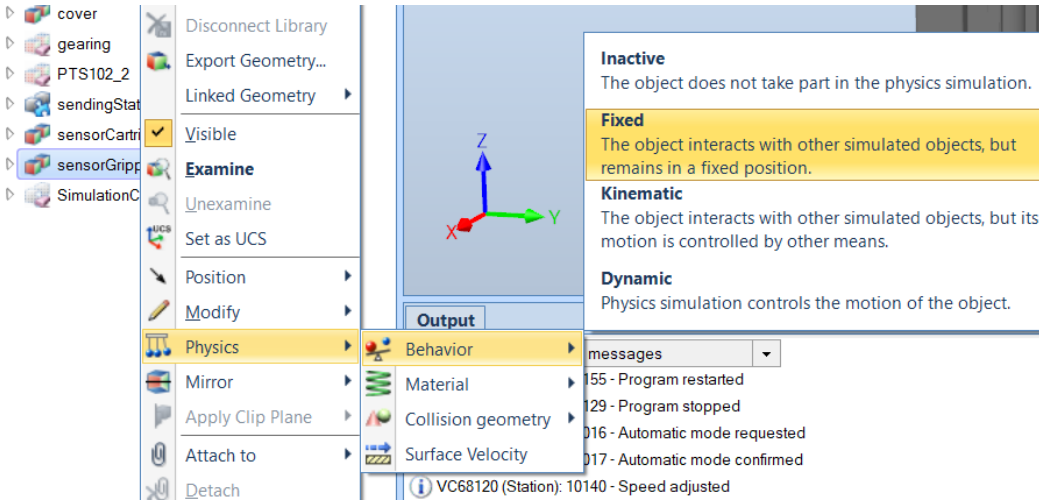
5.3 Přidělení fyzikálních vlastností jednotlivým částem

RobotStudio nabízí možnosti přidělení několika materiálových vlastností simulovanému předmětu. Uživatel si může zvolit druh materiálu mezi již předdefinovanými, popřípadě si nadefinovat vlastní viz obrázek číslo 23. Pro odesílací stanici je ponechána volba oceli.



Obrázek 23: přiřazení vlastností v závislosti na materiálu

Důležitějším nastavujícím parametrem pro tuto simulaci jsou fyzikální vlastnosti samotných komponent přístroje a způsob jejich vzájemného působení mezi sebou. To se nachází v *Physics→Behaviour*. Na výběr jsou celkem čtyři možnosti.



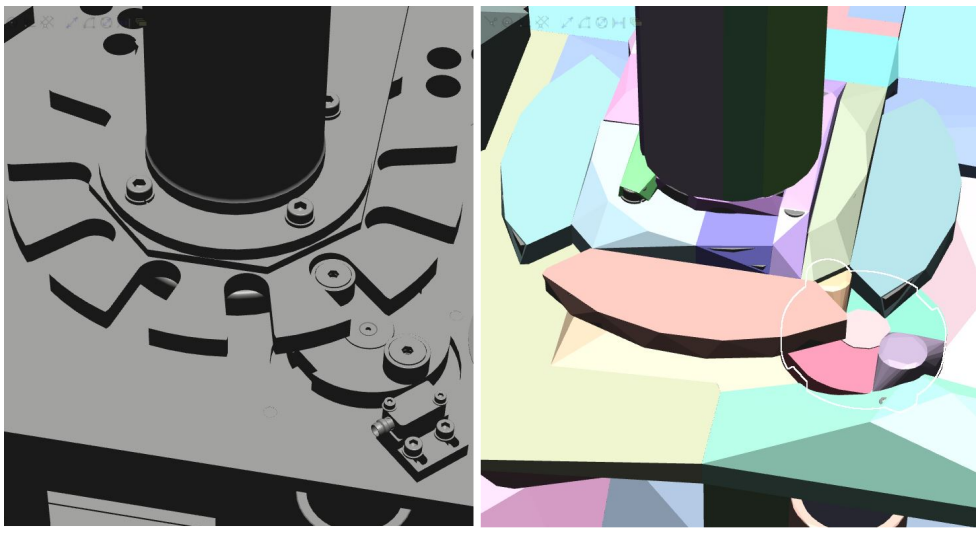
Obrázek 24: výběr fyzikálních vlastností během simulace

- Inactive – Daný objekt během simulace nemá přidělené žádné fyzikální vlastnosti, po spuštění simulace nijak nereaguje na okolní předměty a neřeší se zde kolize mezi sebou.
- Fixed – Objekt má již přidělené fyzikální vlastnosti a může kolidovat s ostatními simulovanými objekty, nicméně má neměnnou pozici. Tuto vlastnost je vhodné přidělit například stěnám.

- Kinematic – V tomto případě má objekt přidělené fyzikální vlastnosti a může kolidovat s ostatními simulovanými předměty. Jeho pohyb je však řízen vnějšími vlivy. Konkrétně se jedná o robotická ramena, či mechanizmy, kde jsou již předdefinované určité pohyby apod.
- Dynamic – Poslední možností je aktivní objekt, který má přidělené všechny fyzikální vlastnosti. Nejen že u něho může dojít ke kolizím, ale daný objekt na kolizi reaguje změnou stavu a ovlivňuje i svoje okolí. Pokud není simulace spuštěna, objekt si ponechává svoje vlastnosti, nemůže však ovlivnit okolní předměty. Toto dodatečné fyzikální chování s sebou také přináší mnohem větší požadavky na výpočetní výkon.

Dle popisu je zřejmé, že pro simulaci odesílací stanice je nejvhodnější použít pro většinu částí právě dynamické vlastnosti. Po přidělení všem důležitým částem již stačí zobrazit kvalitu kolizí pomocí *Physics→Collision geometry→Show collision geometry*. Celý stroj by se měl zabarvit do rozdílných barev, které zároveň budou vyznačovat kvalitu geometrie uplatňující se u simulace. V ideálním případě by nově vzniklá plocha měla kopírovat stávající model.

Po zobrazení kolizí však nastává první problém s příliš nedokonalou kolizní geometrií modelu. Příkladem může být porovnání mechanizmu motoru a ozubeného kola, kde na obrázku číslo 25 je vlevo stanice s vypnutým zobrazením kolizí a vpravo stejné místo se zobrazenými kolizemi. Pro jistotu jsou kontrolovány všechny nastavitelné parametry při



Obrázek 25: příklad problému s kolizemi

exportu a importu modelu v obou programech. Zkoušeny jsou i alternativní metody, kdy je z daného funkčního celku, například ramena, vytvořena knihovna a ta je následně znovu naimportována do simulace. Změnou oproti původní části je to, že nový celek již nelze dělit na jednotlivé součástky, a navenek se tak jeví jako pouze jeden díl. Výsledná kvalita kolizí však zůstává stále stejná.

Tento problém byl přes zákaznickou službu předán české podpoře ABB, která po dodání podkladů měla určit, zda se jedná o limit programu, či zda existuje řešení, které by výslednou kvalitu kolizní geometrie bylo schopno detailněji propracovat. Žádná zpráva o výsledku již však nepřišla. Nezbývá než neoficiálně konstatovat nevhodnost RobotStudia pro simulaci fyzikálních vlastností strojů, kde je kladen důraz na přesnost kolizí v řádech milimetrů.

Původním záměrem simulace bylo řídit elektromotor pomocí signálů z PLC. Za použití dynamických vlastností okolních komponent by následně dokázal rozpohybovat manipulační rameno. Těchto vlastností by se využívalo i u všech pístů a otvírače, který v inicializační části . Chování jednotlivých funkčních celků je tedy třeba dopředu nastavit. Tím se snižuje hodnota celé simulace, neboť stroj teoreticky nemůže přejít do polohy, kterou nemá nikde nastavenou.

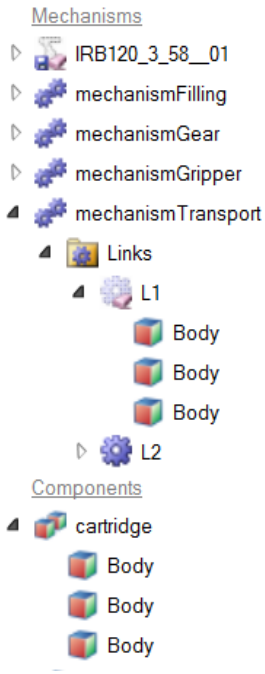
Dále tedy bude používáno nanejvýše kinematické chování u mechanizmů a fixní či neaktivní chování pro zbylé části odesílací stanice. V tom případě již není ani zapotřebí definovat podlahu a postavení stroje.

5.4 Vytváření mechanizmů

Veškeré pohyby stroje budou řízeny přes mechanizmy, do kterých budou zahrnuty skoro všechny části krom patrony s víčkem. Vytváření mechanizmů má určité limity. Je proto vhodné si před samotnou tvorbou rozmyslet, co vše je třeba zahrnout do mechanizmu, jaké vazby mezi sebou budou mít jednotlivé díly a do jaké míry je požadováno provádět na sobě nezávislé pohyby. Zde jsou zmíněny některé vlastnosti, které je třeba vzít v potaz před před vytvářením.

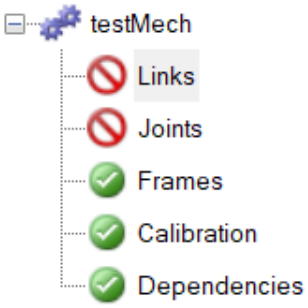
- Jakmile je daná část, například otvírač, zahrnuta do mechanizmu, nelze k ni již přistupovat jako k normální části. Zároveň se už ani nezobrazuje mezi dalšími částmi v levé postranní liště, ale je přesunuta do mechanizmů, kde jí je přidělen název dle čísla Linku v daném mechanizmu.

- V jednom mechanizmu může být použito nanejvýš 7 částí, se kterými lze vytvořit maximálně 6 samostatných pohybů.
- Pokud se veškeré součásti a s nimi zamýšlené pohyby vloží do jediného mechanizmu a zároveň je snahou provádět polohování jednotlivých prvků bez možného ovlivnění polohy dalších částí v mechanizmu, není vhodné využívat funkci, ve které lze předdefinovat přesné polohy a stavy všech částí. Nevýhodou této jinak užitečné funkce je, že uživatel musí definovat polohu všech součástí a nelze vybrat pouze změnu 1 části. To by mohlo vézt k nečekané změně polohy u komponenty, které s tímto pohybem nemá nic společného. Pro tento případ je tedy vhodnější si rozdělit části do více mechanizmů, kde již tento problém nebude.



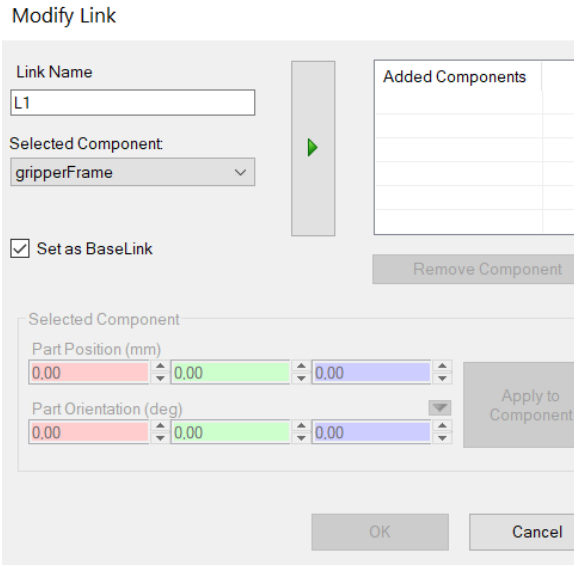
Obrázek 26: části mechanizmu

Po několika pokusech se v tomto případě jeví nejvýhodnější použití více mechanizmů, kde každý bude zajišťovat pohyb 1 komponenty. Typ mechanizmu je volen jako *Device*, což je univerzální typ pro použití u částí bez vazby na robotické rameno. Pro vytvoření je třeba nadefinovat alespoň 2 položky, které se zobrazují s červenou značkou.



Obrázek 27: tvorba mechanizmu

Do položky *Links* je třeba přidat všechny části, které se v mechanizmu budou využívat. Po rozkliknutí se zobrazí tabulka, ve které se vybere první část, v tomto případě *gripperFrame*, a klikne se na zelenou šipku, která danou část vybere. Jedná z částí mechanizmu musí mít zaškrtnutou vlastnost *Set as BaseLink*, která se přiřazuje převážně statickému objektu, který se nebude pohybovat a vůči kterému budou vztaženy polohy ostatních dílů. Následně se obdobným způsobem dodefinují zbylé potřebné komponenty, jen se už u nich nezaškrtává *Set as Baselink*. Do jednoho *Linku* lze vložit více dílů, čehož by se využilo při pohybování s více částmi naráz stejným způsobem. V tomto případě to tedy není zapotřebí.



Obrázek 28: vytváření vazeb v mechanizmu

V položce *Joints* se již definuje vzájemná vazba obou dílů. Jsou zde 3 typy možných pohybů:

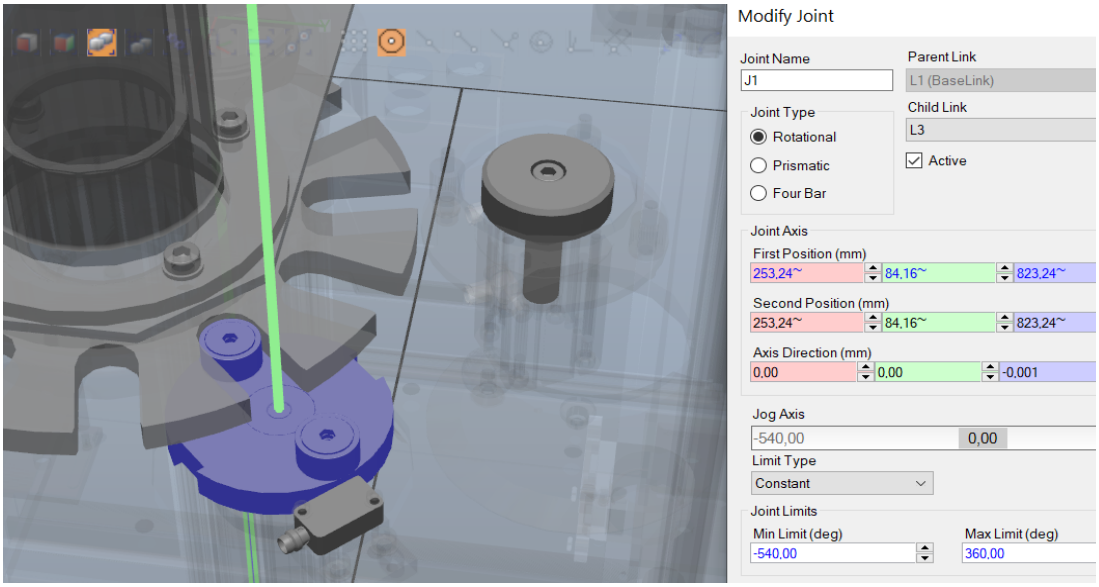
- rotační
- prizmatický (pohyb po přímce)
- *Four Bar* (využití například pro panty u kapoty vozu)

U jednoduchých mechanizmů, kde je definována pouze 1 vazba mezi díly, je *Parent Link* vždy *BaseLink*. Pokud by na *L1* závisel pohyb další části, v druhé vazbě by byl již *L1* definován jako *Parent Link*.

U rotačního pohybu je v porovnání s lineárním třeba určit polohu osy, podél které se natáčení provádí. Na obrázku číslo 31 je zobrazen příklad nastavení. Při záměru rotovat pouze podél osy Z bez dalšího vyosení stačí do kolonky *Axis Direction* pro tuto souřadnici uvézt libovolně velké číslo, kde znaménko určuje směr rotace.

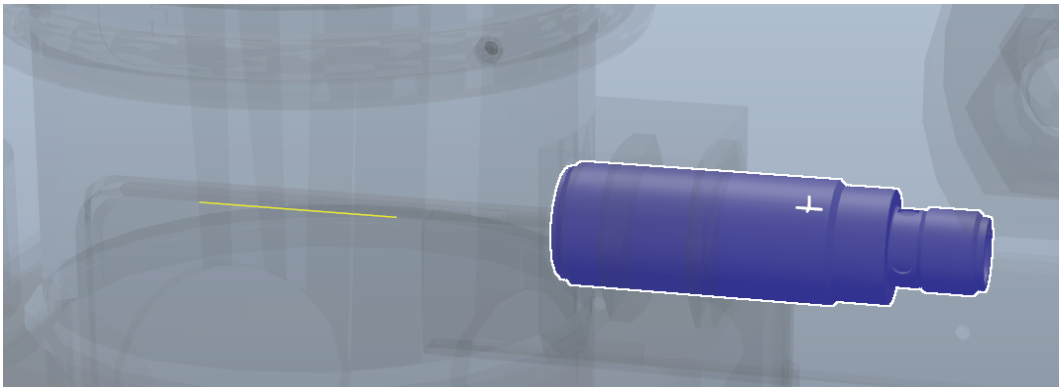
Pří nastavení rozsahu pohybu jsou na výběr tři možnost:

- bez limitů
- dle konstant – uživatel sám stanoví limity, do kterých se objekt může dostat
- dle proměnné



Obrázek 29: vytváření vazeb pro rotační pohyb

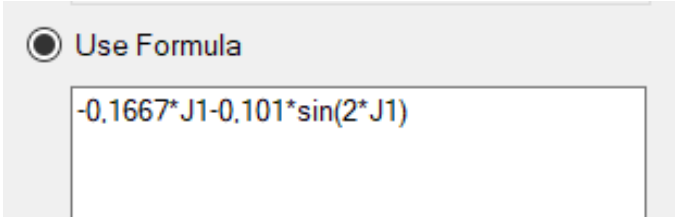
V celé simulaci budou vždy použity limitní typy konstanty. Rozmezí je voleno na základě polohy snímačů krajních poloh. Následné propojení se snímači bude provedeno až později.



Obrázek 30: nastavování krajních poloh v závislosti na poloze snímače

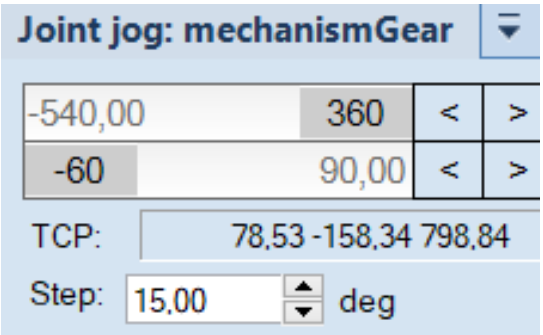
Další položkou mechanizmu, v tomto případě nepovinnou, je *Frames*. Primárně slouží pro nastavení podstavy robotu. Díky ní však také lze v dalších krocích použít funkcí u *Smart Componentů*, která slouží k připevňování předmětů k mechanizmu. Pro její realizaci stačí zvolit *Link*, ke které se vztahuje. V modelu se tento prvek zobrazí pouze jako dodatečný počátek souřadnic.

Poslední užitečnou položkou v nastavení je *Dependencies*, kde je možné vytvořit závislosti mezi jednotlivými již nadefinovanými *Jointy.* Na výběr je možnost zvolení hlavního *Jointu*, podle kterého se s volitelným faktorem mění ten druhý. Pro náročnější požadavky je možné využít matematickou formulaci závislosti, které podporuje i základní goniometrické funkce. Toho je využito u mechanizmu pro natáčení ramene, kde se právě ozubené kolo otáčí v závislosti na motoru tak, aby graficky docházelo k co nejmenším kolizím.



Obrázek 31: použití matematické formulace

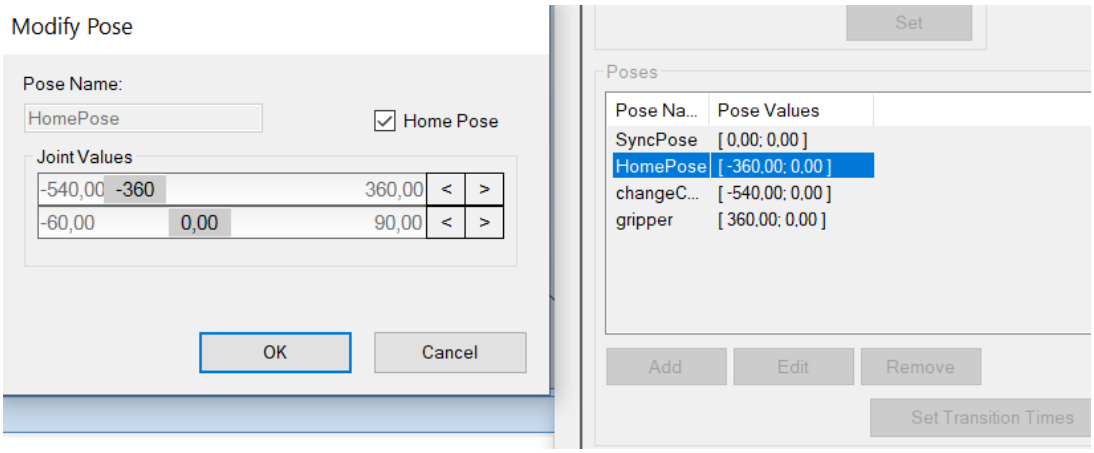
Nyní již lze mechanizmus zkompilovat. Pro jeho ozkoušení stačí na mechanizmus kliknout pravým tlačítkem a vybrat *Mechanism Joint Jog*. V levém dialogovém okně se zobrazí nové možnosti, kde počet os odpovídá počtu nadefinovaných vazeb.



Obrázek 32: testování pohybů

Pokud by se uživatel chtěl vrátit k mechanizmu a upravit ho, je třeba si pouze zkontrolovat, zda k němu není připnuta pomocí *Attach* jakákoliv součást. V tom případě by byla možnost úprav zamčená.

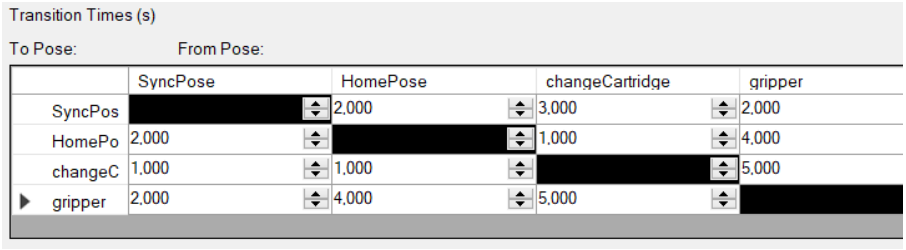
Poslední používanou funkcí u mechanizmů je nadefinování poloh.
Zde je třeba si dát pozor, neboť nadefinování přesných poloh se musí provádět až po zkompilování mechanizmu, v opačném případě se všechny smažou. Dále se doporučuje si celé okno pro vytváření mechanizmu vytáhnout ze základního rozložení stránky, ve kterém tato funkce kvůli špatně nastavenému zvětšování není vidět.



Obrázek 33: předdefinování poloh v mechanizmu

Zde si lze přidat libovolně pozicí, kde se vždy nadefinují polohy všech kloubů. Tato poloha jde poté snadno volat v simulaci přes *Smart Componenty*.

Za zmínku také stojí možnost si přehledně na jednom místě nadefinovat přechody mezi jednotlivými polohami.



Obrázek 34: nastavení přechodů mezi jednotlivými polohami

V tomto případě však bude využit jiný postup a hodnoty uvedené v tabulce nebudou uplatňovány. Vztahují se pouze k použití přes *Move To Pose*, který nelze automatizovat. Tyto přechody je tedy nutno nadefinovat přes příslušný *Smart Component→ PoseMover*, který lze využít i během simulace. Celkem jsou tedy vytvořeny čtyři mechanizmy ovládající pohyb ramene, otvírače, transportního a plnícího pístu.

5.5 Použití Smart Components

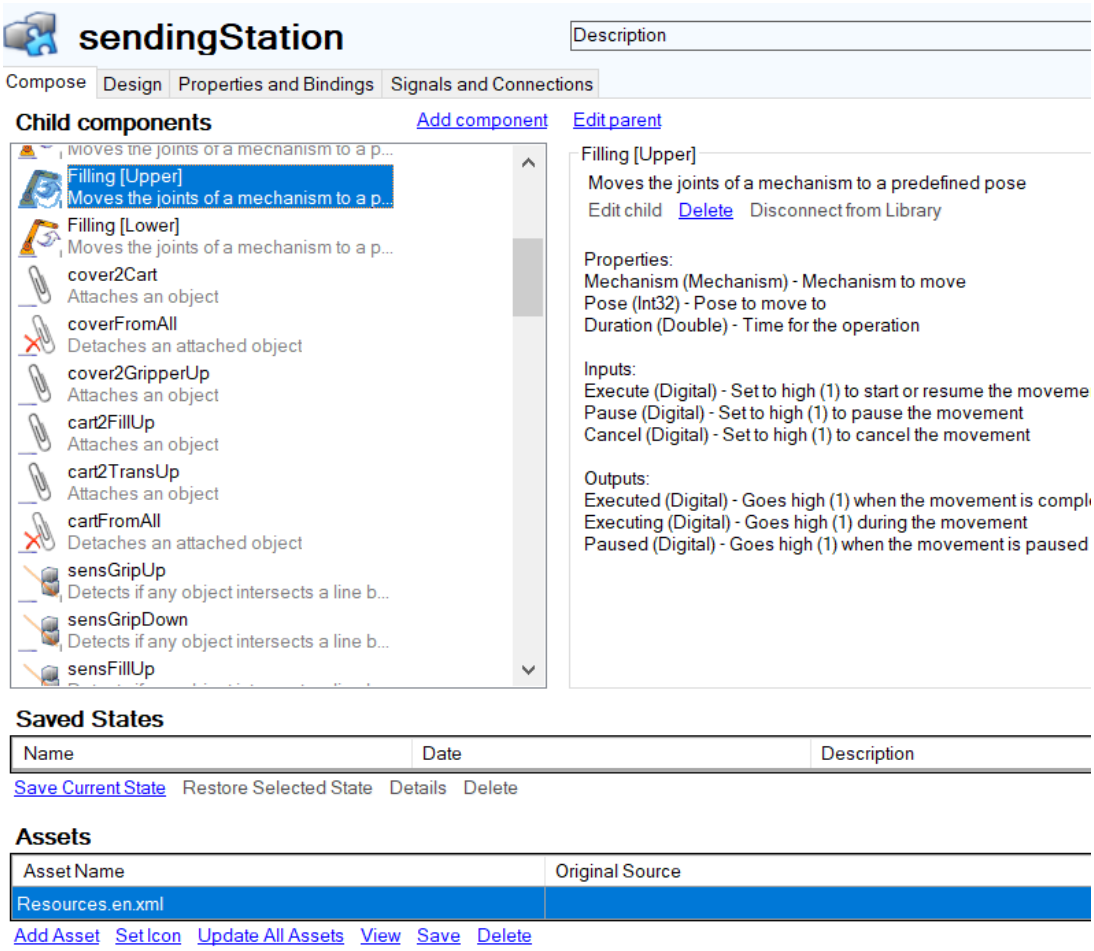
Možnost, jak simulovat veškeré chování předmětného stroje a zároveň toto chování zabalit do jedné struktury, která by se navenek jevila jen jako black box, je za použití *Smart Component*.

V záložce *Modeling* se po kliknutí na *Smart Component* zobrazí nová část tohoto druhu. Po zvolení *Edit Component* se otevře nové okno, ve kterém se nastavuje veškeré chování spjaté s komponentou. Na výběr tu jsou čtyři záložky, které se budou postupně využívat a zaplňovat.

Nejdříve je třeba do sekce *Child components* vložit přes *Add component* prvky zajišťující veškerý pohyb, snímání, připevňování částí k jiným celkům a další.

5.5.1 Okno Compose

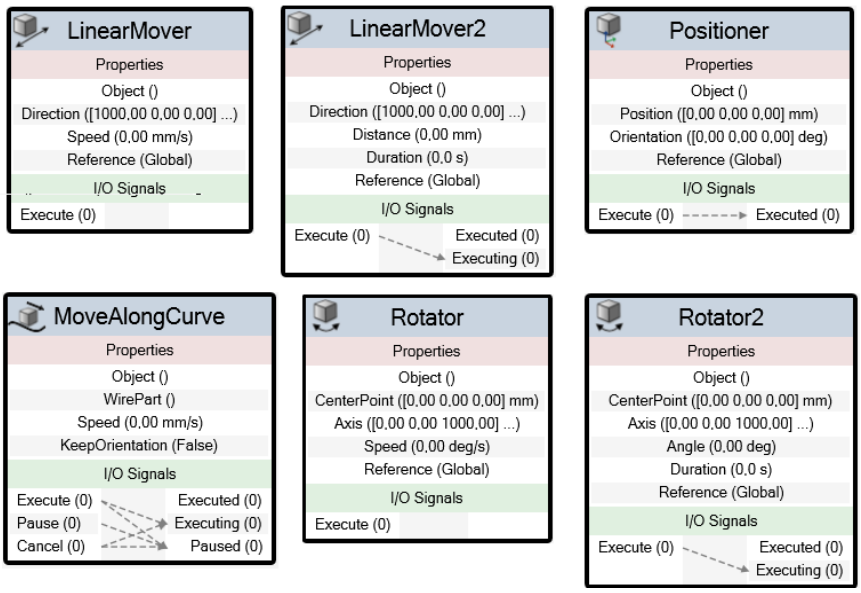
V úvodní záložce se prvně do sekce *Child components* vkládají veškeré prvky zajišťující pohyb, snímání objektů, připevňování částí k jiným celkům a další speciální funkce, které budou postupně ukázány.



Obrázek 35: úvodní okno pro editaci Smart Componentu

5.5.1.1 Přidělení pohybu

V sekci *Manipulators* je výběr všech možných prostředků pro rozpohybovaní dané části či mechanizmu. Některé se svojí funkcionalitou vzájemně překrývají, jiné jsou naopak specifické a mají i různé limity použití. Při jejich zvolení se opět objeví klasické okno s nastavitelnými parametry.

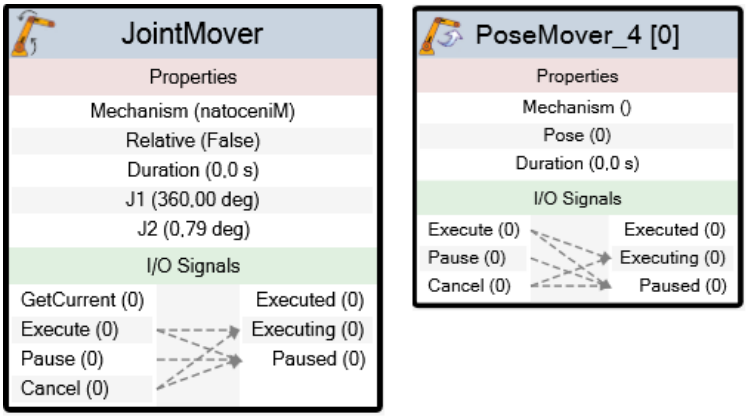


Obrázek 36: rozdíly v jednotlivých Smart Componentech

- LinearMover – Pro pohyb objektu po přímce lze zvolit tuto základní komponentu, kde se nastavuje objekt a směr, kterým se bude pohybovat a jeho rychlost. Dokud je setovaný *Execute*, provádí se povel. Nevýhodou tohoto řešení je absence zpětné vazby ohledně vykonávání.
- LinearMover2 – Přidává definici vzdálenosti, po kterou má být předmětem pohybováno. Navíc obsahuje informaci na výstupu, zda se operace stále provádí a zda je již případně dokončena. Hodnota *Executed* je ve formě krátkého pulzu.
- Positioner – Slouží pouze pro skokovou změnu polohy a pro simulaci je tak nevhodný.
- MoveAlongCurve – Primárně se používá pro simulaci pohybu objektů po dopravníku. Ve stroji není jediný prvek, ve kterém by mělo smysl využít tuto funkci.
- Rotator – Obdoba funkce *LinearMover,* tentokrát však pro rotační pohyby, kde si lze nastavit středovou osu a rychlost otáčení kolem ní. Stejně tak i zde se povel vykonává, dokud je příkaz *Execute* setovaný, a je tedy bez zpětné vazby o stavu.

- Rotator2 – Obdoba funkce *LinearMover2,* oproti *Rotator* se zde nenastavuje rychlost otáčení, ale úhel natočení a doba trvání tohoto úkonu. Opět zde přibyly dvě výstupní proměnné pro informaci, zda se tento blok právě vykonává a zda je již hotový. Pro složitější mechanizmy tu chybí možnost otáčení v závislosti na jiném objektu.

Zbylé dvě možnosti jsou si svojí funkcionalitou zaměřenou na pohyby mechanizmů velmi podobné.



Obrázek 37: bloky pro pohyby s mechanizmy

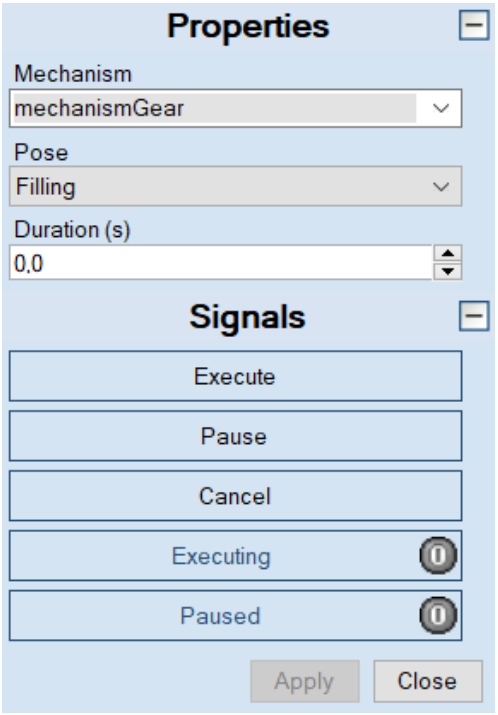
- JointMover – Zde se po vybraní daného mechanizmu zobrazí celkový počet vazeb, se kterými lze pohyboval přes celý definovaný interval hodnot. Do jisté míry je totožný s definicí pozic, která se prováděla po vytvoření každého mechanizmu.

  Výhodou této funkce je možnost zaškrtnutí políčka *Relative*. Díky tomu se tak lze zbavit, při nutnosti hýbat pouze s jedním prvkem z celého mechanizmu, závislosti ostatních hodnot vazeb na této jediné právě používané. Využití *GetCurrent* je především ve významu snímání aktuálních poloh, které lze poslat dál.

- PoseMover – Tato funkce v porovnání s předešlou nabízí stejné vstupní a výstupní parametry z bloku krom *GetCurrent.* Rozdíl je v zadávání požadované polohy, která se zde pouze vybere z uživatelem dopředu nadefinovaných ploch. Navíc zde je výhodou při požadavku na složitější mechanizmy možnost matematicky definovat závislost jedné části na druhé, které je vyžito pro pohon ozubeného kola přes motor.

Jednoznačně nejvhodnější řešení tedy není. Po testování na několika konkrétních příkladech je zvolen postup s prvkem PoseMover.

Při tvorbě se v *Mechanism* vybere již vytvořený mechanizmus a v kolonce *Pose* jedna z předdefinovaných poloh. Aby průběh pohybu nebyl jen skoková změna polohy, je třeba také nastavit *Duration* na plánovanou hodnotu. Stejným způsobem jsou dotvořeny i zbývající pohyby.

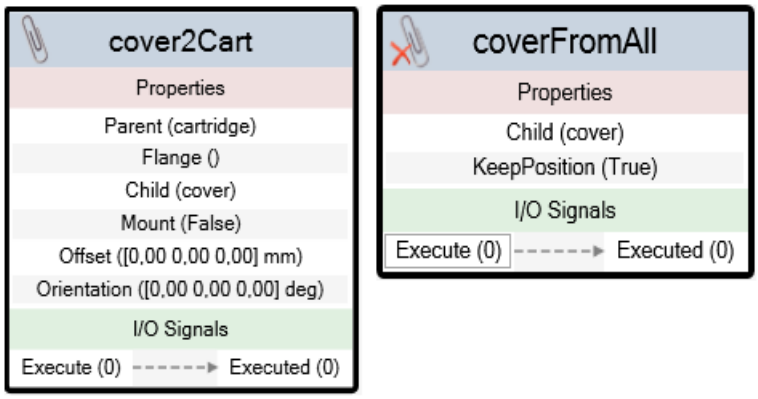


Obrázek 38: GUI pro PoseMover

5.5.1.2 Připevnění částí k sobě

Po rozpohybování veškerých mechanizmů je na řadě manipulace se zbývajícími částmi, které jsou záměrně tvořeny tímto způsobem. Patronu s víčkem je třeba během simulace použít na více místech a nelze ji tak zahrnout do mechanizmů, neboť by mohla být součástí pouze jednoho vybraného.

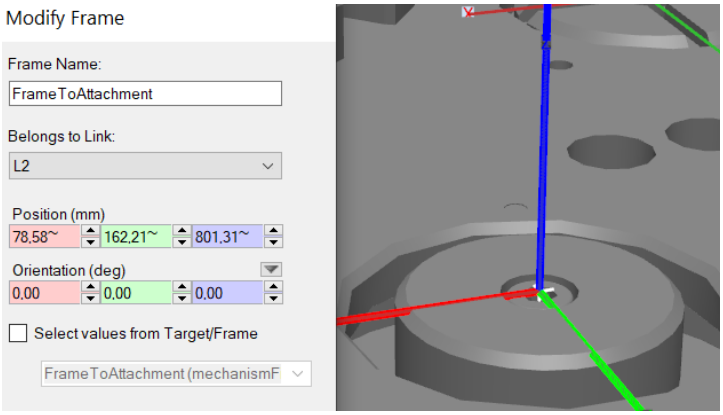
Způsob pohybování, který byl pro tyto dvě části vybrán, je za pomoci funkce *Attach to* a *Detach.* Principem těchto funkcí je připevnění, případně odepnutí od dané části. Pokud je patrona připevněna k jakékoliv jiné části, mechanizmu, či robotu, kopíruje stejnou trajektorii pohybu jako příslušný předmět.



Obrázek 39: bloky k upevnění a odepnutí

Pro připevnění je zapotřebí zvolit rodiče, se kterým je daný potomek spjat. V závislosti na charakteru rodiče může být povinná položka *Flange.* V případě robotu a mechanizmu je tento prvek povinný. U obyčených dílů a *Smart Component* je nepovinný.

Pokud je požadavek na připevnění k mechanizmu, lze tuto smyšlenou, jinak nepotřebnou a neviditelnou část dotvořit v sekci *Frames*. Jedná se pouze o přidání další osy, která by jinak sloužila k podstavě robotu. Nyní již po připevnění patrony k mechanizmu bude patrona kopírovat veškeré změny pohybu, které v mechanizmu nastanou. V minulé kapitole 5.5.1.1 byla využita funkce *PoseMover*, která umožňuje zvolit jen konkrétní pohyb v mechanizmu, jež se má vykonat. Jedná se už o *Smart Component,* a není tedy třeba položku *Frames* během připevňování zadávat. Při testování a ladění však bylo zjištěno, že takto vytvořený pohyb se neprovede a tok dat se u tohoto bloku zastaví.



Obrázek 40: nastavení Framu

Funkce *Attach* také nabízí možnost uchycení přes zaškrtnutí pole *Mount*, při jejimž zvolení jde nastavit, že po připevnění k rodičovi se potomek vycentruje do zvolené polohy. Zde toho není využito.

Opačnou funkci *Detach* je třeba použít při oddělení této vazby. Stejně tak se jedná o povinný mezikrok, který musí byt použit mezi každou změnou rodiče. U této funkce stačí už jen uvézt daného potomka a zda po oddělení má být jeho poloha nezměněna.

V této kategorii je tedy vytvořeno celkem sedm různých připevnění pro patronu a tři pro víčko patrony. Pro oddělení stačí pouze jedna pro každou z částí.

5.5.1.3 Nastavení oblasti snímání

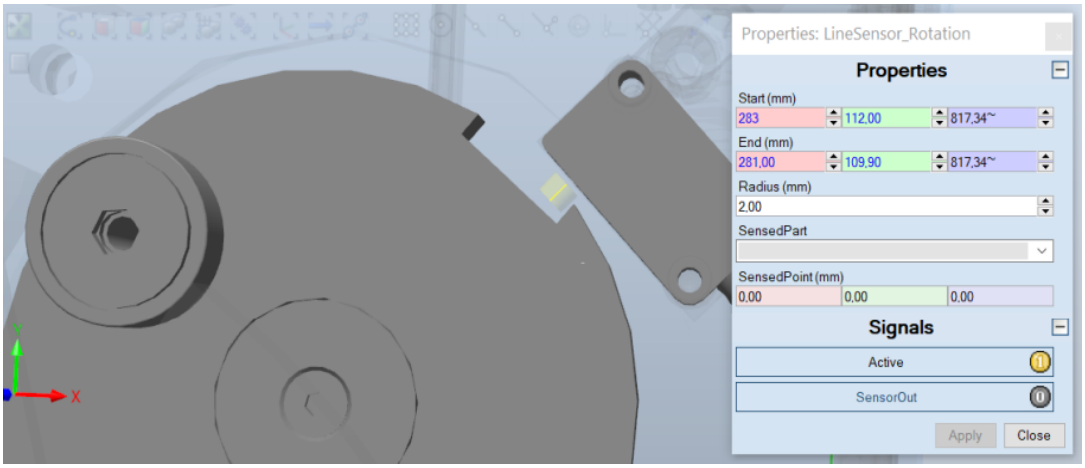
V modelu strojního zařízení je zobrazena většina snímačů polohy a tlaku. Vzhledem k povaze simulace se nemá cenu zabývat snímači tlaku. Pro ověření polohy daných částí jsou již však snímače důležitým kontrolním prvkem. Opět je na výběr více snímačů s různou detekcí.

- CollisionSensor – Dokáže detekovat kolizi dvou definovaných objektů s nastavitelnou tolerancí kolize v milimetrech, které může být případně ignorována. Nicméně v systému není použito dynamické chování objektů, proto zde tento způsob nemá uplatnění.
- ClosesObject – Po zvolení referenčního objektu či bodu v prostoru dokáže zjistit, který další objekt, například robot, a která jeho část, například pracovní nástroj, je nejblíže.

- PositionSensor – Během simulace zobrazuje polohu jistého objektu vůči vztaženému bodu.
- JointSensor – Jedná se o samostatný snímač, který je jinak součástí *JointMover*. Výstupem snímače je aktuální poloha všech vazeb ve zvoleném mechanizmu.
- VolumeSensor – Po definování libovolného objemu senzor vyhodnocuje, název objektu a zda se do tohoto objemu vešel. Případně je možnost nastavit detekci i pouze částečného průniku.
- PlaneSensor – Zde se pro změnu nadefinuje rovina, které snímá protnutí libovolným objektem.
- LineSensor – Pracuje na stejném principu jako dva předešlé, tentokrát se již volí pouze úsečka snímající kontakt s okolními objekty.

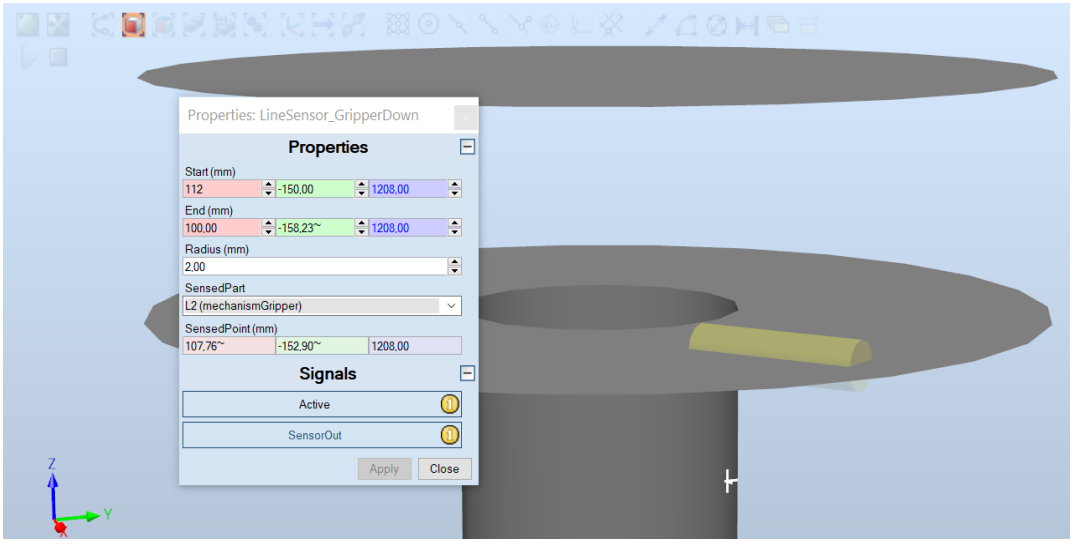
V rámci této simulace jsou nejvhodnější *PlaneSensor* a *LineSensor*. Čistě z praktického hlediska byl pro všechny snímače v modelu vybrán způsob s *LineSensor*, kde se snadněji nastaví snímací oblast.

Na obrázku číslo 41 je příklad nadefinování úsečky s nastavenou šířkou 2 mm. Pokud by se motor ještě o pár stupňů pootočil, v komponentě na snímání by se zobrazil jeho název, přesný bod, kde ke kontaktu došlo, a výstup *SensorOut* by se nastavil do logické jedničky. Pokud není požadováno dostávat informace z výstupu snímače, stačí zakázat jeho činnost přes tlačítko *Active*.



Obrázek 41: ukázka nastavení snímače natočení

Při nastavování je třeba si hlídat vlastnosti modelu neboť jak je ukázáno na dalším obrázku, vnitřek jednoho z pístů je prázdný, a snímač by tak mylně informoval, že píst v dané pozici není. Snímací oblast je tedy nastavena na spodní plochu, kde je pro jistotu i zvětšen poloměr na 2 mm.



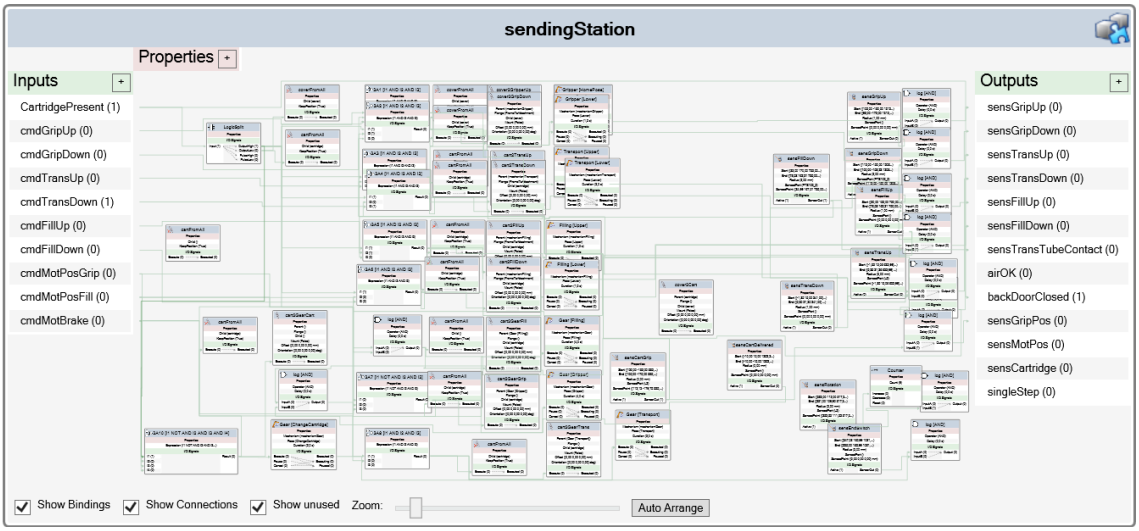
Obrázek 42: nastavení alternativní snímací polohy

5.5.1.4 Signálové funkce

Poslední kategorií, která se v této sekci využívá, je kategorie obsahující převážně logické funkce. Ač je snahou strojnímu zařízení zbytečně nepřidělovat vlastní logiku, když veškeré řízení se provádí přes PLC a stroj by měl plnit pouze mechanickou stránku simulace s výstupy od snímačů, logické prvky zde budou sloužit jako takové dodatečné potvrzení dosaženého stavu. Často je tak použit logický součin AND pro potvrzení polohy dané části, kdy je na vstupu informace od snímače a proměnné informující o dokončení pohybu určitým směrem.

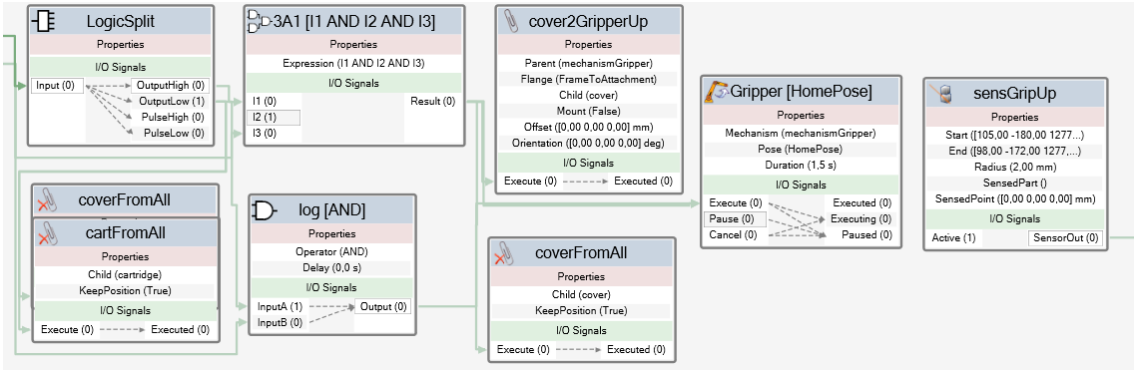
5.5.2 Okno Design

V tomto okně jsou blokově zobrazeny všechny vytvořené Smart Componenty ze záložky *Compose*. Přes postranní prvky *Inputs* a *Outputs* je nadefinován počet vstupů a výstupů, se kterými odesílací stanice pracuje. Na vstupu jsou tedy nadefinovány proměnné odpovídající řídícím signálům z PLC a výstupy z bloku odpovídají seznamu snímačů.



Obrázek 43: průběh při sestavování blokového schéma odesílací stanice

Příklad zapojení pro pohyb otvírače směrem vzhůru je znázorněn na obrázku číslo 44 Kvůli absenci kolizí je zde přidán jeden kontrolní vstup, který informuje o přítomnosti či nepřítomnosti patrony v odesílací stanici. Jakmile je patrona mimo stroj, v bloku *LogicSplit* je aktivní *OutputLow*, který nastaví bloky na zrušení připevnění patrony a víčka od všeho. Pokud je zároveň požadavek na přesun do horní polohy, provede se pohyb samotného otvírače. V opačném případě se přechází do další podmínky s trojitým logickým součinem, kde druhým vstupem je příkaz na přesun otvírače do horní polohy a třetí je zpětná vazba od snímače informující o poloze ramena s patronou na pozici otvírače. Po splnění podmínky se pro jistotu odpojí víčko od případné předešlé vazby a zároveň se připevní k mechanizmu zajišťující pohyb otvírače. Následuje pohyb nahoru, který po dokončení nastaví na svém výstupu *Executed* logickou jedničku.



Obrázek 44: příklad zapojení pro horní pozici otvírače

Tento signál byl původně využíván jako dodatečná zpětná vazba k potvrzení dosaženého stavu, kde se až při logickém součinu s hodnotou snímače posílal na výstup simulovaného stroje signál, že otvírač je opravdu v horní pozici. Nicméně nastával problém během inicializace, kde stroj nemohl navenek informovat o svých stávajících polohách.

5.5.3 Okno Signals and Connections

I/O Signals

Name	Signal Type	Value
sensCartridge	DigitalOutput	0
currentStepDone	DigitalOutput	0
CartridgePresent	DigitalInput	0
sensGripUp	DigitalOutput	0
sensGripDown	DigitalOutput	0

Add I/O Signals Expose Child Signal Edit Delete

I/O Connections

Source Object	Source Signal	Target Object	Target Signal or Property
sensRotation	SensorOut	log [AND]	InputB
sensEndSwitch	SensorOut	log [AND]	InputB
Gripper [Upper]	Executed	log [AND]	InputA
Gripper [Lower]	Executed	log [AND]	InputA
Filling [Upper]	Executed	log [AND]	InputA

Add I/O Connection Edit Delete Move Up Move Down

Obrázek 45: seznam vytvořených signálů a propojení

Tato záložka je víceméně alternativa a může do jisté míry sloužit také jako kontrola zapojení záložky *Design*, kde se zvyšujícím se množství bloku a propojení značně klesá přehlednost. Navíc někdy se nepřekreslují aktivní spojení a na první pohled se daný blok jeví jako nezapojený.

V tabulce *I/O Signals* jsou zobrazeny všechny již nadefinované vstupy a výstupy z bloku *SendingStation*,. Pro vyvedení libovolného vstupu či výstupu vnitřního bloku na vstup či výstup simulovaného celku je k dispozici *Expose Child Signal.* V druhé částí *I/O Connection* se definuje již samotná vnitřní struktura zapojení, kde se vždy vybere zdrojový a cílový blok včetně přesných vstupních a výstupních hodnot, které mají být spolu propojeny.

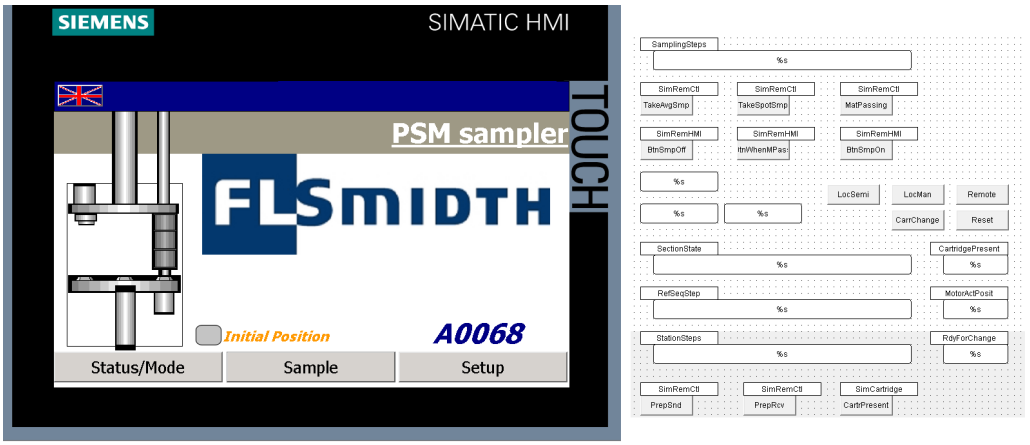
6. PROPOJENÍ MODELU S ŘÍZENÍM

V této kapitole je nejdříve vysvětlena stávající koncepce řízení, ke které jsou postupně doplňovány další nezbytné části k propojení celého řetězce. Jsou diskutovány i alternativní cesty, které při tvorbě byly brány v úvahu.

6.1 Stávající koncepce řízení

Hlavním ovládacím prvkem je SIMATIC HMI TP700 Comfort panel, ve kterém je nahrán program od FLSmidth. Krom možnosti řízení odesílací stanice obsahuje také všechny důležité parametry a stavy v textové či grafické formě včetně chybových hlášení a varování.

Pro základní testování je v TwinCATu vytvořen i program pro simulaci bez nutnosti připojení hardwaru včetně jednoduché vizualizace viz obrázek číslo 48.
Zde je po povolení manuálního řízení možnost spustit program a sledovat průběh stavového automatu simulující proces inicializace a zpracování vzorku materiálu.



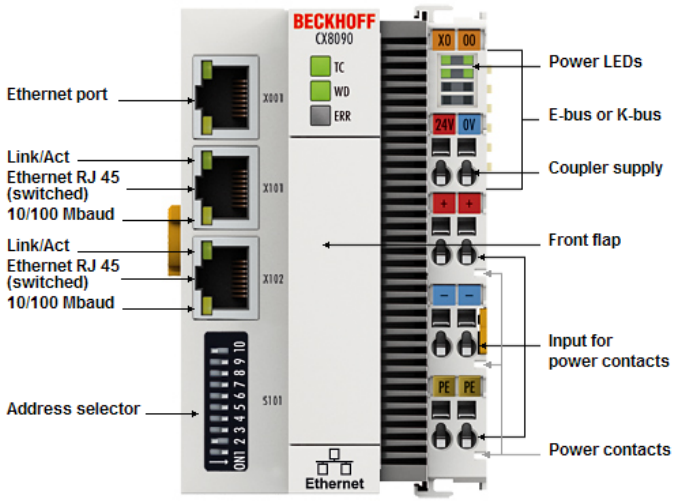
Obrázek 46: dosavadní vizualizace na panelu (vlevo) a v TwinCATu (vpravo)

6.1.1 Řídící PLC

Řídící program stroje PTS 102 je navržen pro PLC Beckhoff CX8090. Aby bylo možné v rámci diplomové práce testovat navrhovaná řešení přímo s tímto typem, bylo od brněnské pobočky Beckhoffu do školní laboratoře zapůjčeno stejné PLC.

Mezi jeho základní parametry patří:

- Procesor: 32 bit, 400 Mhz, ARM9
- Vnitřní hlavní paměť: 64 MB RAM
- OS: Microsoft Windows CE 6.0
- Protokoly: RT-Ethernet, ADS, ModbusTCP, TCP/IP, UDP/IP
- Napájení pro I/O terminály: 2 A (max) [8]
- a další viz obrázek číslo 47



Obrázek 47: použité PLC CX8090 s popisem[8]

Pomocí přepínačů ve spodní části lze napevno přidělit adresu PLC.

6.1.2 TwinCAT 2

Kvůli doposud nulovým zkušenostem s produkty Beckhoff byla pro bližší seznámení se a pochopení systému TwinCAT využita nabídka na čtyřdenní programátorský kurz u brněnské pobočky. Součástí zadání je práce s programem psaným ve vývojovém prostředí Beckhoff TwinCAT2. Dnes se už jedná o zastaralé prostředí, které bylo dávno nahrazeno novějším a komplexnějším nástupcem TwinCAT3. Co se kompatibility týká, TwinCAT 2 32b je možné provozovat nanejvýš na OS Windows 7 a Windows 10 IoT Enterprise. Ač je zde i alternativa ve formě 64 bitové verze, které pracuje i pod Windows 10, runtime je možné používat jen s 32 bitovým OS[9].

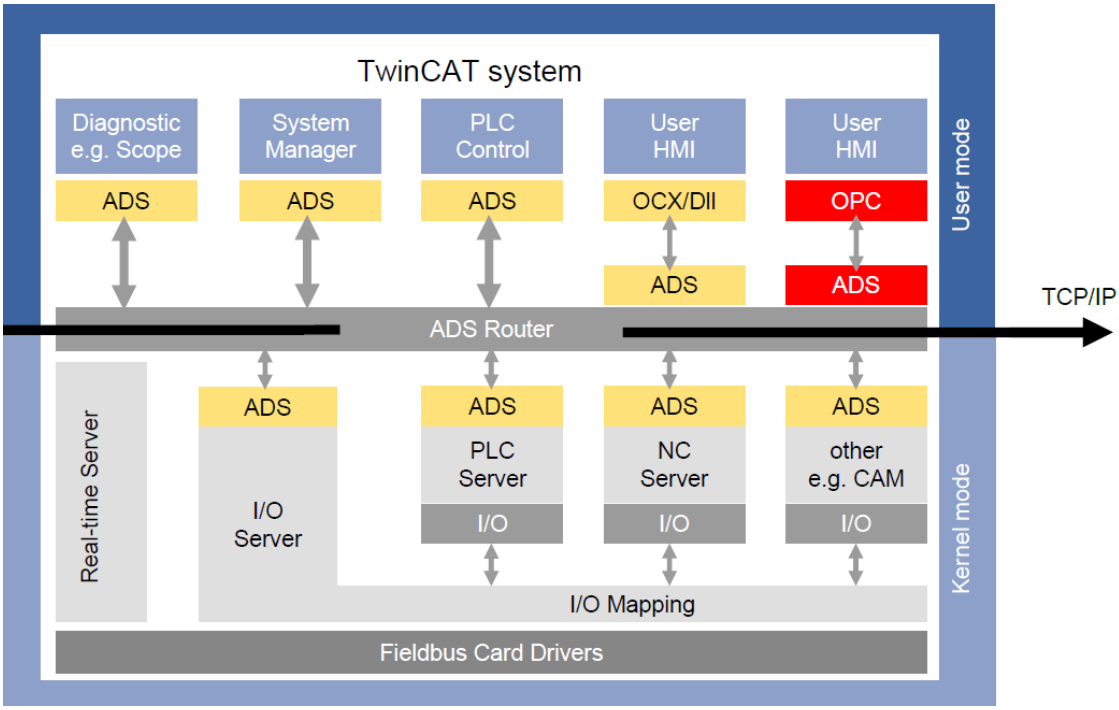
Prostředí v sobě zahrnuje 2 základní části.

- TwinCAT PLC control (programování)
- TwinCAT System manager (HW konfigurace)

Je možné používat všechny jazyky dle standardu IEC 61131-3.

- IL (Instruction List)
- LD (Ladder Diagram)
- FBD (Function Block Diagram)
- ST (Structured Text)
- CFC (Continuous Function Chart)
- SFC (Sequential Function Chart)

Komunikace probíhá pomocí ADS, což je otevřený komunikační protokol vyvinutý firmou Beckhoff, který běží nad TCP/IP a využívá principu klient-server.



Obrázek 48: architektura systému TwinCAT2[9]

6.2 Komunikace mezi PLC a RobotStudiem

Při výběru způsobu propojení simulovaného stroje s PLC byly brány v úvahu dvě možné cesty.

6.2.1 Použití C#

První z možností je vytvoření API v jazyku C#. Příkladem může být práce Denise Fraipointa, který celé řešení zahrnul do Smart Componentu a je volně dostupné i přímo přes *RobotStudio Apps*. Jeho aplikace je postavená na knihovnách Snap7, což je open source multiplatformní ethernetová komunikační sada pro nativní propojení s PLC Siemens S7. Komunikace mezi Snap7 a virtuálním PLC probíhá přes TCP/IP a vyžaduje ještě další program s názvem NetToPLCSim. Ten přidělí virtuálnímu PLC lokální adresu, k niž se tento Smart Component může připojit.[10]

6.2.1.1 Propojení s RobotStudiem

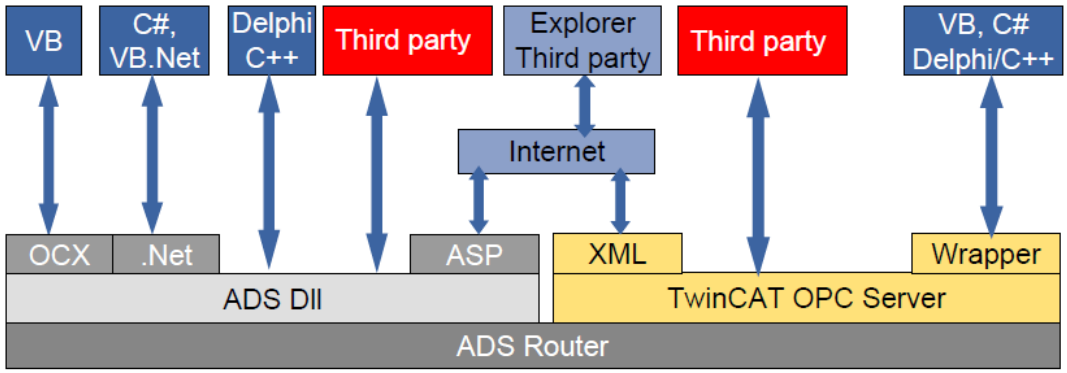
Pro tvorbu C# programu je k dispozici mnoho knihoven, kde například *ABB.Robotics.Controller.PC* umožňuje přes reálný i virtuální kontrolér přistupovat k proměnným obsažených v RAPID kódu.

Další možnost možností je použít předpřipravenou šablonu pro vytvoření vlastní Smart Componenty, kterou lze následně naimportovat do RobotStudia a používat ho jako kterýkoliv jiný Smart Component.

6.2.1.2 Propojení s TwinCATem

Součástí ADS je také ADS API, které programátorovi pro práci s různými platformami nabízí knihovnu s názvem *TwinCAT.Ads*. Té lze využít i na systémech Windows CE.

Tento přístup byl ze začátku brán jako výchozí, nicméně postupně se přešlo na druhý způsob bez použití C#.[11]



Obrázek 49: komunikace mezi C# a ADS [9]

6.2.2 Použití TCP/IP socketu

Pro přímé komunikační spojení mezi TwinCATem a RobotStudiem je možné využít knihovny se základními funkčními bloky k TCP/IP komunikaci, které Beckhoff nabízí. Zde je tedy zapotřebí vytvořit stranu serveru a klienta, kde simulace v RobotStudiu bude mít charakter serveru a PLC se jako klient bude dotazovat na jeho aktuální stav.

6.3 Implementace na straně serveru

V následujících podkapitolách jsou zmíněny všechny nezbytné kroky vedoucí k propojení mezi simulací odesílací předmětného stroje a TCP/IP serverem psaném v RAPID kódu.

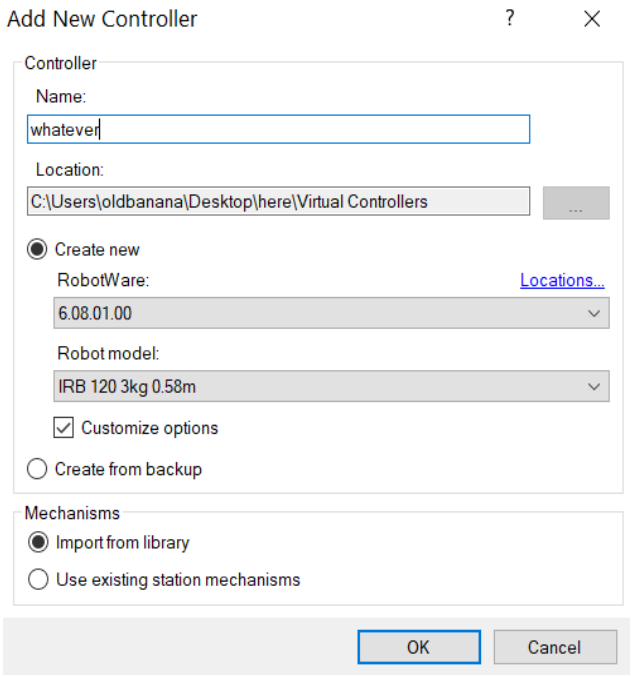
6.3.1 Vytvoření a nastavení virtuálního kontroléru

Pro vytvoření serveru a následné propojení se simulovaným strojním zařízením je již zapotřebí kontroléru. Ten je možné vybrat či vytvořit přes *Home→Virtual Controller→New Controller*. V tomto případě byl kontrolér vytvořen ještě ve starší verzi RobotStudia s Robotware 6.0.8. Od verze 2019 je možné využít i RobotWare s verzí 7. Pro tyto účely použití si lze vybrat jakoukoliv z možností, neboť postačí pouze základní vlastnosti, které jsou u obou verzí totožné. V dialogovém okně je na výběr také typ robotu. Opět se jedná o libovolnou položku. Důležitým krokem je zaškrtnutí políčka *Customize options*. Po potvrzení se následně otevře nové dialogové okno s rozsáhlými možnosti úprav.

Mimo speciální funkce pro robotická ramena, pohony a průmyslové sítě je zde záložka *Communication*, ve které nutné vybrat *616-1 PC Interface*. To poskytuje rozhraní mezi kontrolérem robotu a počítači připojenými v síti. Je nezbytné pro TCP/IP komunikaci, SDK uživatelské aplikace, které přistupují k datům kontroléru, a další. Při použití RobotWare verze 7 je již tato funkce součástí a není zapotřebí ji dodatečně nastavovat. [6]

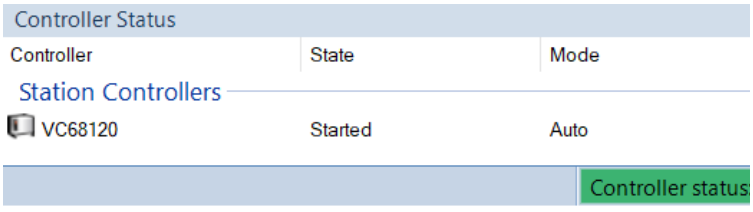
V záložce *Engineering Tools* je položka *628-1 Sensor Interface*, která se však váže na reálné snímače v tomto případě ji nelze využít. Pro využití více úloh naráz je potřeba zvolit možnost *623-1 Multitasking*. Díky tomu lze oddělit hlavní pohybovou úlohu a například úlohu pro TCP/IP komunikaci, která při volbě typu úlohy *Semistatic* je spuštěna automaticky na pozadí a zastavení jiných úloh na ní nemusí mít vliv.

Po vybrání všech potřebných komponent a přesunutí do poslední záložky se už jen zobrazí kompletní seznam pro kontrolu. Posledním krokem je potvrzení a následně se již nově vytvořený Virtuální kontrolér zobrazuje v nabídce dostupných.



Obrázek 50: vytvoření nového kontroléru

Pro vložení Virtuální kontroléru existuje i druhý způsob přes záložku *Controller→Add Controller→Start Virtual Controller.* Zde je však třeba upozornit, že při vkládání tímto způsobem se v levé liště zobrazí v kategorii *Virtual Controllers*, nikoliv *Current Station* a v pravém spodním rohu se zároveň neobjeví informace o statusu kontroléru.



Obrázek 51: informace o stavu virtuálního kontroléru

Problém nastává především při snaze spojit ve *Station Logic* bloky se simulovaným strojem a kontrolérem pro komunikaci s PLC, neboť se tento blok ani nezobrazí, a signálově tak nejde propojit.

Při použití virtuálního kontroléru není zapotřebí přes záložku *Configuration→ Communication* nijak nastavovat hodnoty v *IP Setting*, či v *Transmission Protocol*. Ty slouží pouze pro reálný kontrolér.

6.3.2 Struktura přenášených dat

Data jsou přenášena ve formě byte streamu, kde velikost jednoho rámce je 22 bytů. Rámec je rozdělen na hlavičku o velikosti 1 byte, která poskytuje informace o velikosti odesílaných dat, a na samotná data, kde je celkem 21 jednobytových hodnot se všemi stavy snímačů odesílací stanice a s řídícími signály od PLC. Ač RobotStudio dokáže pomocí základních funkcí pro práci se sokety zjistit délku rámce automaticky, pro TwinCAT je třeba tuto hodnotu explicitně nadefinovat. Jelikož jsou všechna data pouze jednobytová, není třeba žádné další zarovnání dat na určitou velikost, které by bylo například při použití čtyřbytových hodnot nutné.[6]

Z celkového počtu 21 binárních hodnot jich slouží 11 jako zpětná vazba od snímačů a 1 byte je přiřazen také pro případnou volbu jednokrokového ovládání. Zbylých 9 bytů jsou řídící výstupy z PLC pro manipulaci s písty, otvíračem, motorem a brzdou.

6.3.3 Kód v RAPIDu

Pro vytvoření programu v jazyku RAPID je zapotřebí se přepnout do oken *Controller* či *RAPID*, po kterém se již změní obsah levé lišty. Po rozkliknutí záložky *RAPID* je vidět předpřipravená úloha s názvem *T_ROB1*. Do ní je třeba vložit nové moduly. Ty představují sady deklarovaných dat a sady rutin. S moduly lze pracovat jako se samostatnými soubory, které je možno ukládat, načítat a kopírovat.

Zde je na výběr mezi 2 typy modulů:

- Program – Během vykonávání lze moduly přidávat a odebírat. Přehledně zobrazuje všechny použité procedury.
- System – Používají se k definování běžných dat a rutin specifických pro systém. Příkladem mohou být nástroje. [6]

V tomto případě je zvolen programový modul. Žádné dodatečné vlastnosti pro omezený přístup a použití již nejsou použity.

Kód v RAPIDu je rozdělen do 2 modulů, kde *TcpServer* slouží k zajištění TCP/IP komunikace se vzdáleným klientem a k předání nezpracovaných dat modulu *DataProcess*. Ten data po zpracování přiřazuje na výstupy bloku *VC68120* a zároveň do proměnných nahrává hodnoty ze vstupů bloku.
Data jsou přijímána v raw bytech, tudíž je třeba ručně přiřadit jednotlivé datové byty k příslušným signálům. Část systémových modulů je ponechána v základním nastavení.

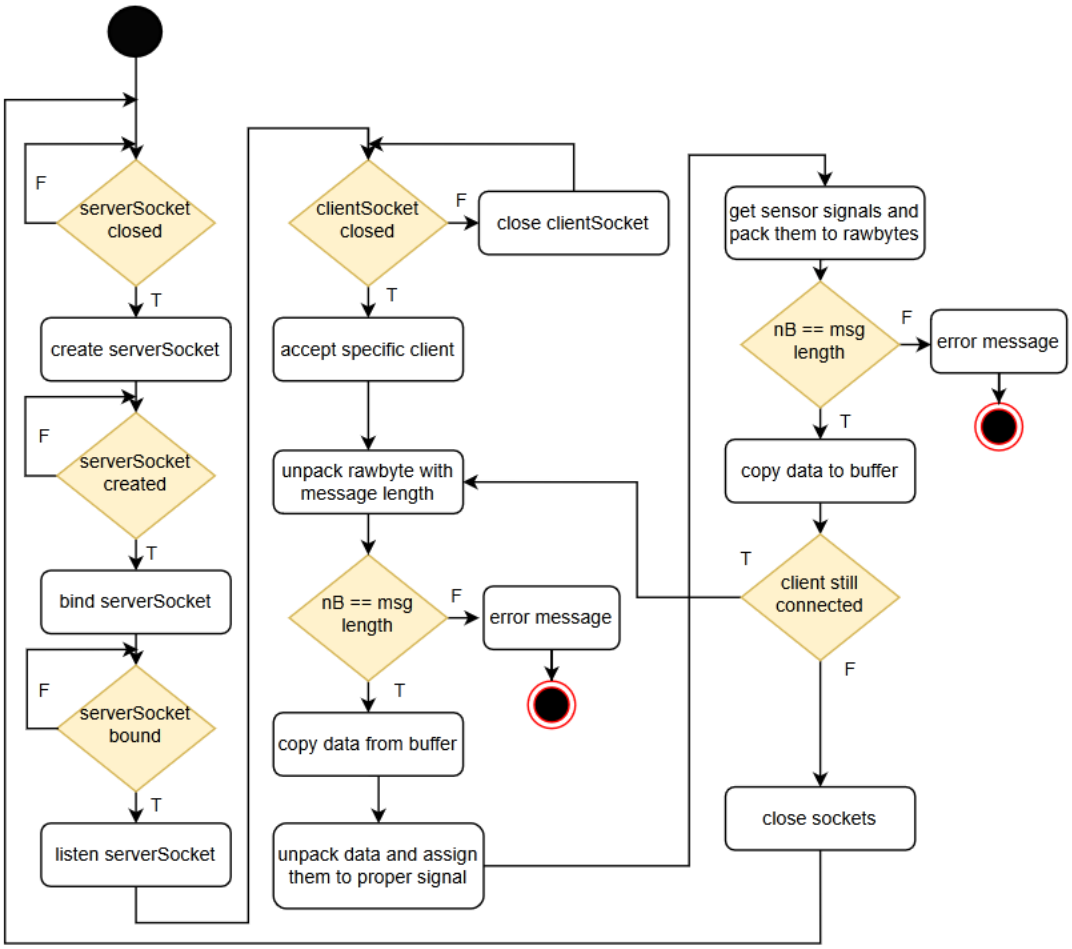
6.3.3.1 TcpServer module

Na začátek jsou deklarovány a definovány globální proměnné, včetně struktur reprezentujících formát zprávy, seznam signálů patřících ke snímačům a seznam příkazů od PLC. Pro vytvoření struktury slouží klíčová slova *RECORD* a *ENDRECORD.*

Stěžejní procedurou v tomto modulu je *main()*, který postupně volá všechny podprocedury. První z nich je *socketInit()*, která slouží ke kontrole stávajícího stavu serverového socketu a k vytvoření nového, na kterém bude server poslouchat. Následuje *waitingForClient()*ve kterém probíhá ještě kontrola stavu klientského socketu a následně již server akceptuje požadavek na připojení od klienta se specifickou IP adresou, v tomto případě 192.168.1.10.

Nyní se přechází do smyčky *WHILE*, ve které probíhá výměna dat mezi oběma stranami do doby, než je připojení ze strany klienta ukončeno. Zde se nejdříve přijímají data od vzdáleného klienta. Ta jsou ve formě raw bytů načtena do zásobníku, odkud se postupně rozbalují byty na požadovaných pozicích. Dle prvního bytu, v tomto případě číslo uložené v hlavičce, se kontroluje shodnost počtu uvedeného v hlavičce a počtu opravdu přijatých dat od klienta. Pokud se hodnoty liší, program se ukončí s chybovou hláškou informující o neshodě. Jinak se data přepošlou do programového modulu *DataProcess*, odkud se následně přijmou zabalená data s aktuálními hodnotami od stroje. Pokračuje se opět v *TcpServer* modulu, kde se v podproceduře *sndMsg()* přiřazují příslušná data na správnou pozici do zásobníku. Před odesláním opět probíhá kontrola na požadovaný formát zprávy a následně se již data posílají ke klientovi.

Pokud je klient stále připojen, pokračuje se znovu ve výměně dat. V opačném případě jsou sockety uzavřeny a přechází se na začátek k inicializaci nového spojení.



Obrázek 52: diagram aktivit pro TCP/IP server program

Během vykonávání celé procedury se přes chybové funkce kontroluje timeout a zda v průběhu programu nebyl nečekaně uzavřen socket. V tom případě se program snaží vytvořit nové spojení. Momentálně je všude doba čekání nastavena na maximální možnou, tudíž nyní chyba timeoutu nemůže nastat.

6.3.3.2 DataProcess module

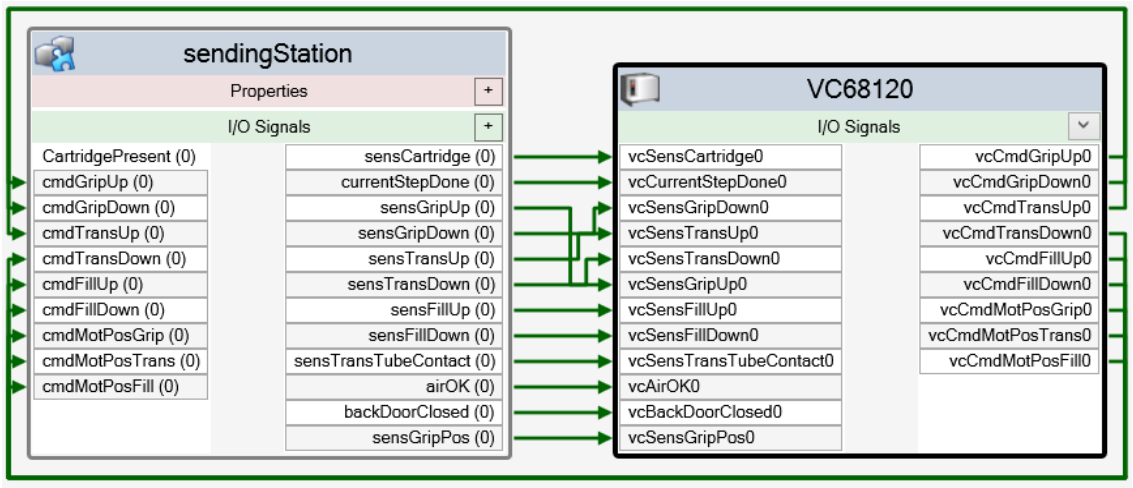
Úkolem tohoto modulu je především zjednodušení a zpřehlednění *TcpServer* modulu, od kterého přijímá data v raw bytech. Balík dat obsahuje veškeré řídící signály z PLC a zpětné vazby od snímačů odesílací stanice. Pokud jsou moduly použity ve stejné úloze, v tomto případě *T_ROB1,* proměnné obou modulů jsou navzájem viditelné. Výjimkou je deklarace uvnitř funkcí a procedur.

6.3.4 Napojení TCP/IP serveru na simulovaný stroj

Samotné napojení je realizováno ve *Station Logic*. V záložce *Design* je následně propojen blok *sendingStation*, zahrnující veškeré chování odesílací stanice, a blok *VC68120*, realizující serverovou stranu TCP/IP komunikace a zpracovávající data do formátu pro komunikaci se *sendingStation.* Data předávaná mezi bloky jsou pouze v binární formě.

Přidání libovolného vstupu či výstupu u *sendingStation* lze pomocí ikony + v pravém rohu bloku. U kontroléru je již nutné mít tyto signály dopředu nadefinované například přes záložku *Controller→Configuration→Add signal*, kde se následně zobrazí dialogové okno.

V případě bloku *sendingStation* lze kdykoliv vstupní i výstupní proměnnou upravit či odstranit. U kontroléru je nutné odstranit vazbu na tuto proměnnou, zavřít okno *Station Logic* a a následně restartovat kontrolér. Tato proměnná však nadále zůstává uložena v *Configuration→I/O System→Signals.*



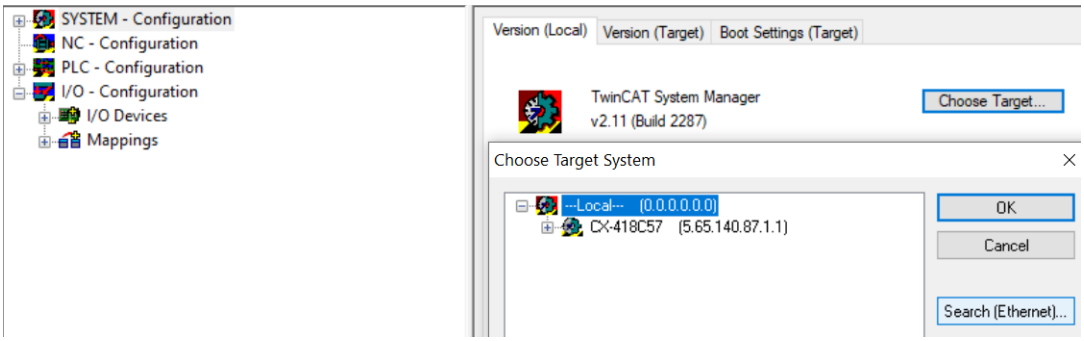
Obrázek 53: propojení odesílací stanice s kontrolérem

6.4 Implementace na straně klienta

Do fungujícího TwinCAT 2 programu je tedy třeba doplnit část realizující klientskou TCP/IP komunikaci včetně bufferu. Přenášeno by mělo být 21 globálních proměnných, které jsou jinak namapovány na vstupy a výstupy reálných karet. Ty však nejsou součástí zadání.

6.4.1 Propojení s PLC a nahrání programu

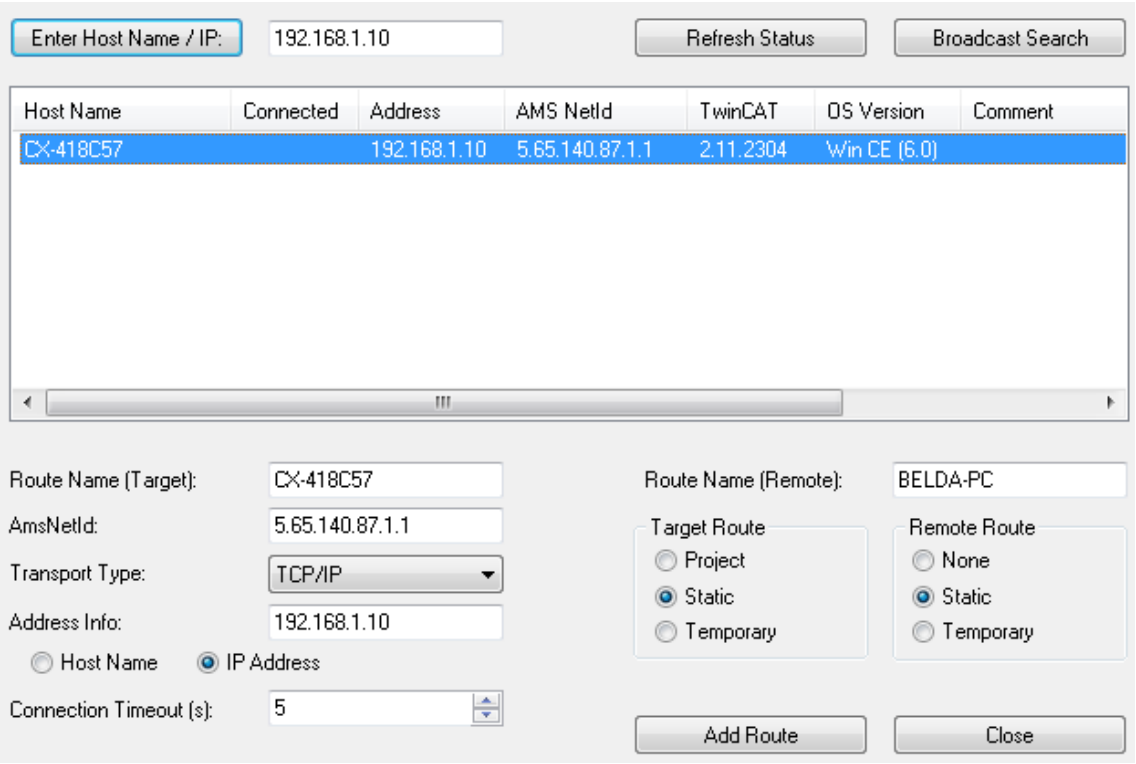
Pro spojení s PLC je nutné nejdříve spustit *System Manager,* který slouží především pro hardwarovou konfiguraci, a přejít na záložku *SYSTEM – Configuration,* kde v bočním okně je tlačítko *Choose Target…* a následně v dalším dialogovém okně zvolit možnost *Search(Ethernet)…*



Obrázek 54: postup k propojení se s PLC

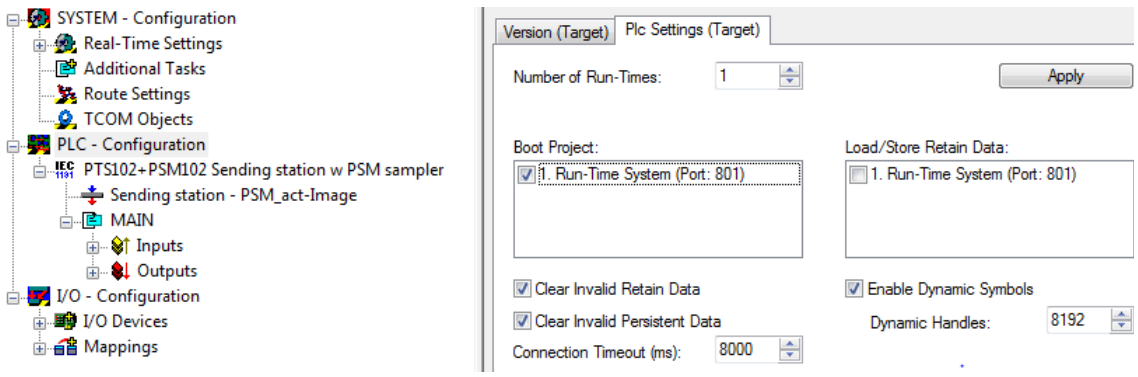
Po jeho zvolení se zobrazí okno sloužící pro vyhledání zařízení viz obrázek číslo 57. Vyhledání zařízení může být provedeno dvěma způsoby. Pro znalosti přesné IP adresy PLC lze využit levé horní okno na zadání a následně stisknout *Enter Host Name / IP.* Druhou možností je využití tlačítka *Broadcast Search* přes které se vyhledají všechna dostupná zařízení. Před vyhledáváním je doporučeno si v levém spodním rohu zaškrtnout možnost *IP Address*.

Jakmile je zařízení nalezeno, je možné se sním spojit přes tlačítko *Add Route*, při kterém se uživatel přihlásí názvem účtu a heslem. Po úspěšném propojení se ve sloupci *Connected* objeví X. Další nastavení v tomto okně nejsou třeba. V předešlém okně se nyní vybere již aktivní PLC.



Obrázek 55: vyhledávání a propojení PLC

Nově se poté na spodní liště System Manageru zobrazuje připojené PLC s AMS Net Id a jeho stav. Následně se v záložce *PLC – Configuration* přidá požadovaný program a v pravém okně v *Plc Settings(target)* se nastaví požadovaný počet runtimů včetně dodatečných vlastností.



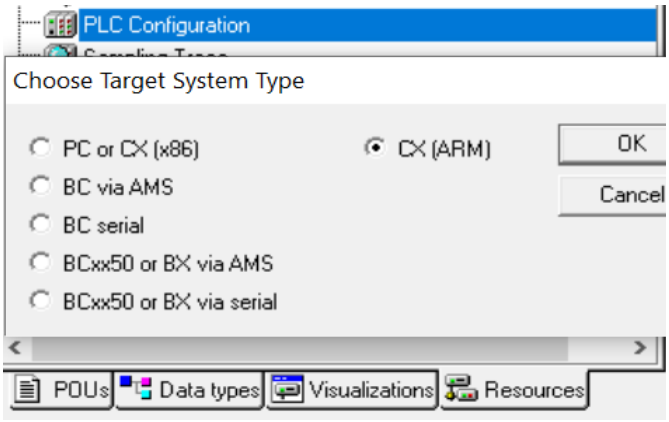
Obrázek 56: nastavení runtimů

Poté se už jen z horní lišty vybere možnost *Activate Settings* a v zápětí se zvolí možnost přepnutí TwinCATu do režimu *Run*. Všechny události a včetně chyb jsou zobrazovány ve spodní části System Manageru.

Vše potřebné je nastaveno a lze nyní přejít do druhé části s názvem *PLC Control* věnující se už samotnému programu a jeho struktuře. Ten je dělen do čtyř hlavních záložek.

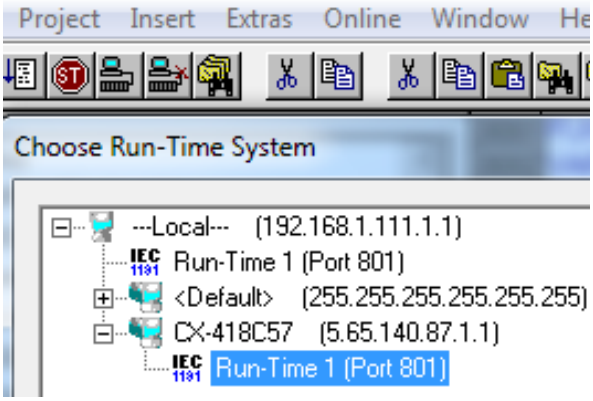
- POUs – Zde jsou umístěny všechny programy, funkční bloky a funkce.
- Data types – Slouží pro definice struktur a vlastních typů.
- Vizualization – V této záložce jsou případné vizualizace. Nachází se zde i původní testovací vizualizace k simulaci.
- Resources – Tato záložka obsahuje všechny globální proměnné, seznam použitých knihoven a správce na případné přidání dalších. Dále je zde logger informující o všech událostech, monitorování signálů, alarmy, watch tabulka a nastavení programu dle typu PLC.

Pro počáteční nahrání projektu je třeba v *PLC Configuration* zkontrolovat nastavení programu. Pro model CX8090 je nutné natavit *CX(ARM)*.



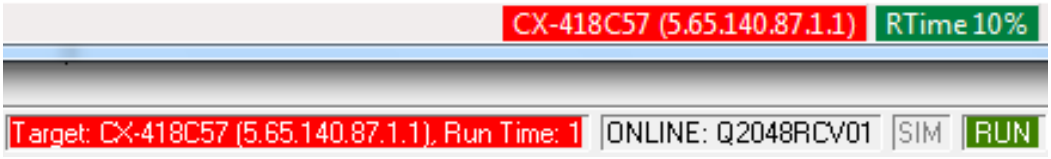
Obrázek 57: výběr správného typu PLC

Následně již stačí přes *Online→Choose Run-Time System...* zvolit přednastavený runtime z PLC. Nyní je už možné přes *Login* nahrát program do PLC a tlačítkem *Run* spustit program a sledovat aktuální hodnoty.



Obrázek 58: výběr správného runtimu

O Stavu je uživatel opět informován přes spodní lištu okna viz obrázek číslo 59.



Obrázek 59: zobrazení stavu v System Manegeru (nahoře) a PLC Control (dole)

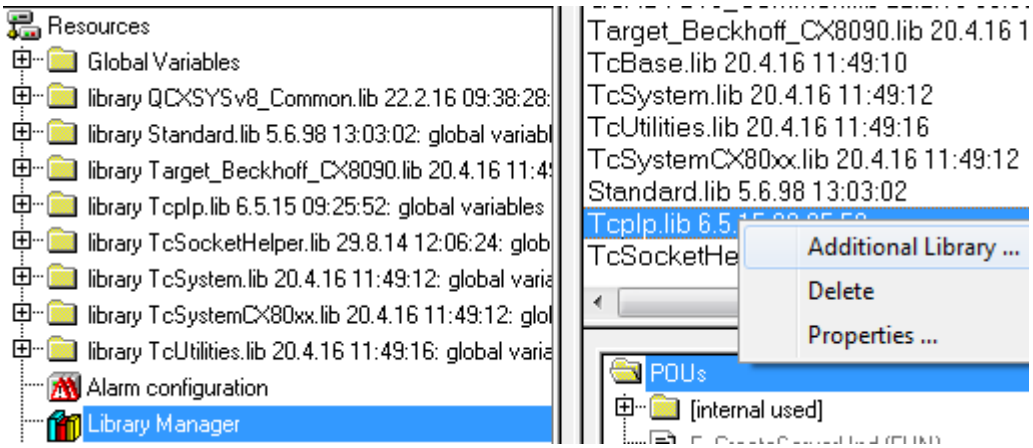
6.4.2 TCP/IP funkční bloky

Ke správné funkci je krom naimportovaných knihoven třeba mít také na pozadí spuštěn proces *TwinCAT TCP/IP Connection Server*. Pro klasické operační systému typu Win32 či Win64 je k dispozici zdarma demo licence, pro systémy CE již není demo verzi možné využít. Přes firmu FLSmidth byla tedy zajištěna plná verze, aby bylo možné pracovat s PLC.

V případě, že je PLC program tvoří klientskou stranu komunikace, je pro minimální funkcionalitu třeba implementovat alespoň tyto bloky:

- *FB_SocketConnect* a *FB_SocketClose* k vytvoření a uzavření komunikace se vzdáleným hostitelem (tyto 2 funkce také lze nahradit jedinou *FB_ClientServerConnection*, která zapouzdřuje chování obou zmíněných bloků
- FB_SocketSend a FB_SocketReceive pro datový přenos se vzdáleným serverem[12]

Po doinstalování TCP/IP pro TwinCAT je nutné naimportovat nové knihovny. To se provádí přes záložku *Resources→Library Manager*, kde se následně v pravém okně vybere možnost *Additional Library…,* poté již stačí dohledat umístění knihovny na disku a knihovna se stává součástí projektu. Ve spodním okně si lze prohlédnout, co vše daná knihovna zahrnuje.



Obrázek 60: nahrání dodatečných knihoven do projektu

6.4.3 Tvorba programu

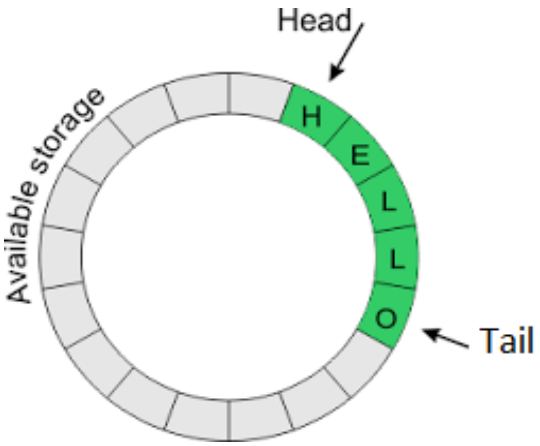
Program zajišťující TCP/IP komunikaci je realizován v jazyku ST. Základem pro tuto aplikaci byla ukázka funkčnosti jednotlivých bloků, kterou na svých stránkách výrobce nabízí. Program pro klientskou stranu je opět rozdělen do několika částí.[12]

6.4.3.1 FB_DataBuffer

Vyrovnávací paměť pro zápis a čtení hodnot je v programu řešena přes kruhový buffer. Ten má charakter fronty typu FIFO, což znamená, že první zapsaná informace do paměti je následně také jako první přečtena.

U kruhového zásobníku jsou aktivní 2 ukazatele, kde první ukazuje na adresu, kam se mají data zapsat (Tail), a druhý ukazuje na pozici, odkud se naopak budou data vyčítat (Head). Pokud se vloží nový prvek do paměti, automaticky se o 1 zvýší i index ukazatele pro čtení hodnoty. Pokud je již zaplněn i poslední prvek pole a zároveň první je už přečtený, začíná se s přepisováním paměti od prvního indexu.

Velikost paměti je nastavena na 100 bytů, což při velikosti odesílaného rámce 22 bytů bez problému postačuje. [13]



Obrázek 61: princip ring bufferu[14]

6.4.3.2 FB_Client

Tento funkční blok slouží k vytvoření klienta, kde na vstup jsou přivedeny informace o serveru, ke kterému se má připojit, a také povolení samotného klienta. Jelikož se klient nepřipojuje k dalšímu TwinCAT serveru, položka s *AmsNetID* zůstává prázdná a stačí pouze definovat IP adresu a port.

Klient dále volá funkční blok pro realizaci připojení k serveru s názvem *FB_Connection.* Pokud je úspěšně připojen, volá další blok *FB_DataProc* pro práci s daty. Výstupem z klientského bloku je stav připojení vůči serveru.

6.4.3.3 FB_DataProc

Zde probíhá zpracování dat, kde vstupně-výstupními parametry jsou hodnoty pro odeslání a přijmutí uložené v zásobníku.

Dle časovače jsou při zápisu všechny požadované proměnné vloženy do struktury k odeslání a přidány na konec fronty. V případě přijímání se hodnoty ze zásobníku načtou do jiné struktury pro příjem dat od serveru, přičemž zároveň probíhá kontrola, zda byly po přečtení úspěšně odstraněny z fronty.

6.4.3.4 FB_Connection

Logikou se jedná o nejsložitější blok. Pracuje se stejnými vstupně-výstupními parametry jako *FB_DataProc* a zbylé vstupní výstupní parametry se shodují s blokem *FB_Client.* Vnitřní chování je rozděleno do třech stavů.

- Připojování – V tomto kroku jsou použity nastavené hodnoty k připojení serveru a probíhá pokus o připojení. Jakmile se klient úspěšně připojí, vyčistí se hodnoty uložené v zásobníku a čítače pro přijatá a odeslaná data a přechází se do dalšího stavu.
- Výměna dat – Zde se již řeší odesílání a přijímání dat od serveru. Po kontrole připojení a zaneprázdněnosti se v části odesílání dat nejdříve smažou nejstarší data. Pokud vše proběhne v pořádku, do datového rámce se uloží data a do hlavičky se načte údaj o délce zprávy. Následně se zavolá funkce na odeslání zprávy a tím sekvence pro odesílání končí. Po stejné úvodní kontrole se kontroluje i obsah volného místa v paměti a následně se volá blok pro přijmutí dat. Jakmile jsou nějaká data přijata, kontroluje se, základní nastavení časovače pro přijímání se nastaví na 0, aby mohl data zpracovat co nejdříve. Během následující kontroly se zjišťuje shodnost obdrženého počtu dat s očekávaným a po splnění se začíná kopírovat do paměti. Pokud po určitou dobu neobdrží klient data od serveru, přechází se do stavu odpojení.
- Odpojení – Pokud se klient dostane do tohoto stavu, po uplynutí určitého času se znovu snaží připojit.

7. TESTOVÁNÍ CELÉHO ŘEŠENÍ

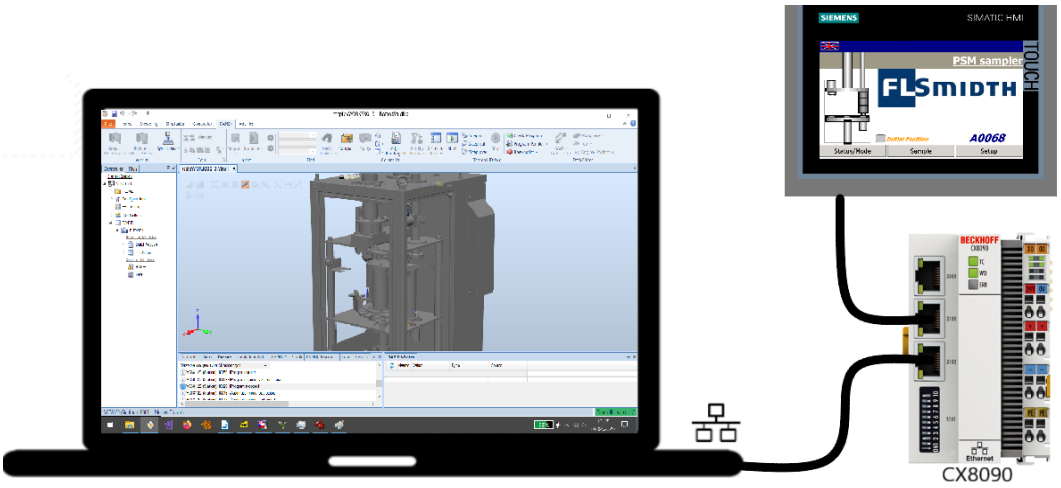
Tato kapitola popisuje propojení celého systému, jeho závěrečného testování a doladění včetně závěrečného shrnutí výsledků.

7.1 Komunikace mezi programy TwinCAT a RobotStudio

Testování funkčnosti TCP/IP komunikace během vývoje bylo nejdříve prováděno přes localhost na stejném PC z prostředí TwinCAT 3. Důvodem byla především nutnost spouštět TwinCAT 2 pod OS Windows 7 32bit, který běží virtuálně přes technologii Hyper-V. Zároveň není možné na tomto virtuálním operačním systému používat RobotStudio, neboť má kvůli ochraně licencování velmi omezenou funkčnost.

7.2 Komunikace mezi PLC a RobotStudiem

Po odladění chyb TCP/IP komunikace mezi klientem a serverem bylo již možné testovat spojení přes PLC. Dle elektrodokumentace od FLSmidth jsou PLC a HMI panelu přiřazeny IP adresy odpovídající běžnému provozu. Přes přepínací páčky na PLC je napevno nastavena IP adresa 192.168.1.10 s maskou podsítě 255.255.255.0. Do portu X101 na PLC je připojen HMI panel s adresou 192.168.1.20 a do portu X102 je s adresou 192.168.1.100 připojen počítat s RobotStudiem.



Obrázek 62: zapojení pro testování funkce

7.3 Spuštění a průběh simulace

Simulaci v RobotStudiu je nutné spustit přes okno *Simulation→Play*, při čemž se zároveň spustí i RAPID program. Opačné spouštění simulace z *RAPID→Start* není vhodné. Simulace se sice spustí zároveň s kódem, nicméně dané změny poloh mezi jednotlivými stavy se provádí pouze skokově.

Po nahrání a spuštění programu v PLC se klient automaticky dotazuje na přítomnost serveru. Jakmile je simulace v RobotStudiu spuštěna a klient je připojen, začíná výměna dat. V RobotStudiu o stavu komunikace a přenosu dat informuje okno výstupu, u klienta je informativní pouze výstupní okno v systémovém manažerovi, tudíž už během zapojení s HMI panelem, PLC a RobotStudiem nelze monitorovat stav připojení z pohledu klienta. Pokud na některé ze stran nenastane chyba, probíhá komunikace mezi PLC a RobotStudiem po celou dobu spuštění simulace.

Během poslední fáze se již pouze testují přenášená data aby:

- příkaz z virtuálního kontroléru odpovídal příkazu vykonávaného v PLC.
- na příkaz z virtuálního kontroléru se provedl správný pohyb v simulaci stroje.
- hodnoty snímačů v simulaci byly ve správném pořadí čteny proměnnými v PLC programu.

7.4 Shrnutí práce

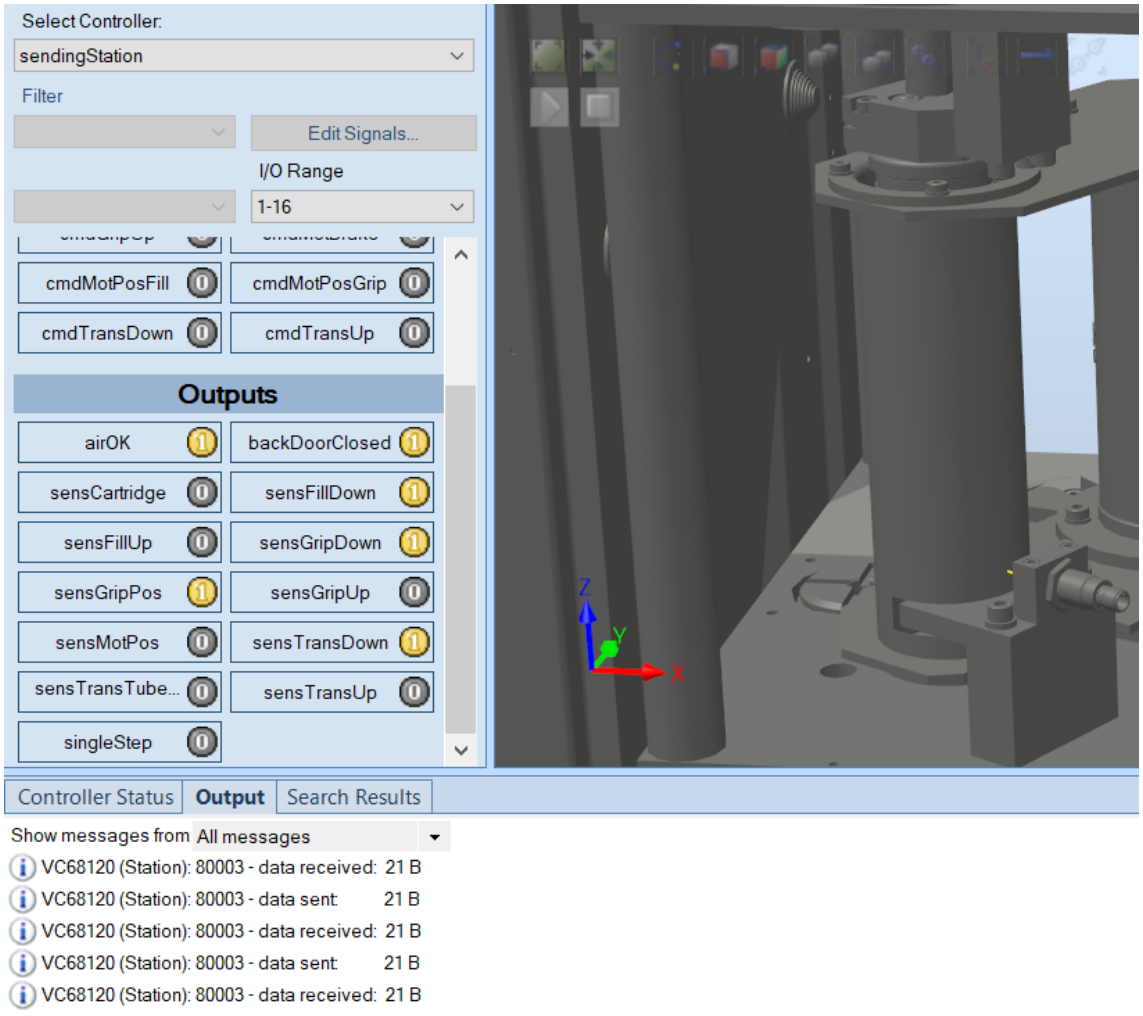
Úkol, zda by tento způsob mohl být do budoucna použit při vývoji nového stroje, prokázal nevhodnost prostředí RobotStudia pro tyto účely. Nemožnost použití dynamických vlastností částí stroje tak značně zredukovala hodnotu simulace. Nicméně vznikla funkční simulace odesílací stanice ovládané přes PLC, kterou lze použít pro demonstraci základního chování.

Mezi návrhy na zlepšení je zařazeno:

- doplnění HMI vizualizace o možnost vidět stav komunikace se serverem včetně přidání základních příkazů pro správu klientské strany, neboť do této části se již nijak nezasahovalo.
- použití logických operací k bitovému zakódování hodnot snímačů a PLC příkazů do několika bytů.

5.686518	192.168.1.10	192.168.1.100	TCP	76 49652 → 201 [PSH, ACK] Seq=507 Ack=507 Win=32612 Len=22
5.692800	192.168.1.100	192.168.1.10	TCP	76 201 → 49652 [PSH, ACK] Seq=507 Ack=529 Win=63250 Len=22
5.890627	192.168.1.10	192.168.1.100	TCP	60 49652 → 201 [ACK] Seq=529 Ack=529 Win=32590 Len=0
5.910631	192.168.1.10	192.168.1.100	TCP	76 49652 → 201 [PSH, ACK] Seq=529 Ack=529 Win=32590 Len=22
5.933080	192.168.1.100	192.168.1.10	TCP	76 201 → 49652 [PSH, ACK] Seq=529 Ack=551 Win=63228 Len=22
6.091793	192.168.1.10	192.168.1.100	TCP	60 49652 → 201 [ACK] Seq=551 Ack=551 Win=32568 Len=0
6.139133	192.168.1.10	192.168.1.100	TCP	76 49652 → 201 [PSH, ACK] Seq=551 Ack=551 Win=32568 Len=22
6.159793	192.168.1.100	192.168.1.10	TCP	76 201 → 49652 [PSH, ACK] Seq=551 Ack=573 Win=63206 Len=22
6.291544	192.168.1.10	192.168.1.100	TCP	60 49652 → 201 [ACK] Seq=573 Ack=573 Win=32546 Len=0
6.363719	192.168.1.10	192.168.1.100	TCP	76 49652 → 201 [PSH, ACK] Seq=573 Ack=573 Win=32546 Len=22
6.369739	192.168.1.100	192.168.1.10	TCP	76 201 → 49652 [PSH, ACK] Seq=573 Ack=595 Win=63184 Len=22
6.496725	192.168.1.10	192.168.1.100	TCP	60 49652 → 201 [ACK] Seq=595 Ack=595 Win=32524 Len=0
6.616632	192.168.1.10	192.168.1.100	TCP	76 49652 → 201 [PSH, ACK] Seq=595 Ack=595 Win=32524 Len=22
6.630517	192.168.1.100	192.168.1.10	TCP	76 201 → 49652 [PSH, ACK] Seq=595 Ack=617 Win=63162 Len=22
6.797075	192.168.1.10	192.168.1.100	TCP	60 49652 → 201 [ACK] Seq=617 Ack=617 Win=32502 Len=0
6.840866	192.168.1.10	192.168.1.100	TCP	76 49652 → 201 [PSH, ACK] Seq=617 Ack=617 Win=32502 Len=22
6.846353	192.168.1.100	192.168.1.10	TCP	76 201 → 49652 [PSH, ACK] Seq=617 Ack=639 Win=63140 Len=22

Obrázek 63: záznam komunikace mezi PLC a simulací v programu WireShark



Obrázek 64: spuštěná simulace s informacemi o přenosu dat a výstupech

8. ZÁVĚR

Cílem práce bylo použití ABB RobotStudia na vytvoření simulace stroje firmy FLSmidth na zpracování vzorků sypkého materiálu a připojení na hotové řízení stroje napsané v prostředí Beckhoff TwinCAT2.

Úvodem byl nastíněn princip virtualizace strojů, kterého je využito i v rámci této diplomové práce, jeho přednosti a uplatnění v průmyslu.

V prvním bodě zadání bylo představeno ABB RobotStudio, které sloužilo k vytvoření simulace předmětného stroje. Důraz byl kladen především na seznámení se s prvky, které byly využity pro tvorbu simulace strojního zařízení a pro komunikaci s PLC. V tomto případě bylo netradičně použito RobotStudio pro simulaci, ve které nebylo využíváno jediné robotické rameno, proto byly krátce zmíněny i funkce související s roboty, které tvoří základní pilíře celého programu.

Další část byla věnována představení samotného stroje na zpracování sypkého materiálu, který byl v RobotStudiu simulován. Základně byly popsány všechny důležité funkční celky, stejně tak pracovní cykly a možnosti ovládání, mezi kterými lze přepínat. Kapitola sloužila především pro lepší pochopení stroje, u kterého bylo snahou všechny tyto vlastnosti přenést do simulačního prostředí.

Následovala kapitola realizace virtuální kopie odesílací stanice v RobotStudiu. Zde byl popsán postup tvorby od prvotního nastavení prostředí až po dokončení simulace a propojení celého komunikačního řetězce. Během vytváření byly zmiňovány limity jednotlivých řešení a případné alternativní postupy. Příkladem limitů byl původní záměr s přidělením dynamických vlastností většině částí v přístroji. Díky tomu by byl pohyb manipulačního ramene stroje řízen přes převod elektromotorem, který by jako jeden z mála komponent dostával řídící signály na změnu polohy od PLC. Písty by přepravovaly patronu na příslušná stanoviště a otvírač by byl schopen oddělit víčko od patrony. Kvůli příliš výrazným kolizím geometrie při simulaci stroje s dynamickými vlastnostmi byl tento postup řešen i s podporou ABB. Zpětná vazba, zda tedy existuje nějaké řešení na odstranění problému, však již nepřišla, a neoficiálně tedy byla konstatována nevhodnost RobotStudia pro simulaci tohoto druhu, kde je kladen důraz na přesnost v řádech milimetrů.

Všechny pohyby tak byly předělány do mechanizmů, které mají své předdefinované koncové polohy a příchodem určitého signálu se do těchto poloh dostanou. Dynamické vlastnosti zbylých součástí byly změněny na neaktivní. Výstupní simulace již tedy neměla tak vysokou vypovídající hodnotu, neboť jednotlivé části se de facto nemohly dostat mimo dopředu definované body.

Způsob komunikace mezi PLC a RobotStudiem byl realizován přes TCP/IP soket, kde RobotStudio představovalo server a PLC klienta. Data byla posílána ve formě byte streamu, kterým se přenášely všechny povely od PLC a zpětná vazba od snímačů z RobotStudia. Po odladění komunikace již probíhalo řízení přes PLC a HMI panel, při čemž se také testovalo správné provedení pohybu při volbě manipulace s jednotlivými částmi stroje.

Výsledkem je tedy model předmětného stroje realizovaný v RobotStudiu, který lze řídit a monitorovat pomocí PLC. Původní záměr, zda by se tento způsob mohl do budoucna uplatnit během fáze vývoje nového stroje, se sice ukázal jako nepoužitelný, nicméně vznikla možnost vizuálně prezentovat chod stroje v závislosti na požadovaném kroku.

Literatura

[1] Automa: *časopis pro automatizační techniku* [online]. Děčín: Automa - časopis pro automatizační techniku, s.r.o., 2016, 22(5). [cit. 11.05.2020] Dostupné z: http://automa.cz/page-flip/casopis/automa/2016/05/index.html#page/40

[2] Techmagazín.cz. *Techmagazín.cz* [online]. Copyright © 2010 [cit. 11.05.2020]. Dostupné z: http://www.techmagazin.cz/45649

[3] SIEMENS. 2019. *Viruální zprovoznění*. [online] Copyright © 1996 [cit. 11.05.2020]. Dostupné z: https://www.industryforum.cz/virtualni-zprovozneni-meni-budoucnost-prumyslu.

[4] Virtual Commissioning and Software Testing - JPro. *JCommerce | Agile software development & IT services company* [online]. Copyright © 2020 JCommerce Sp. z o.o. [cit. 11.05.2020]. Dostupné z: https://www.jcommerce.eu/jpro/articles/virtual-commissioning-software-testing

[5] RobotStudio – silný nástroj pre online i offline programovanie, Exkluzívne články, Rubriky, ATP Journal - priemyselná automatizácia, robotika a informatika. *ATP Journal - priemyselná automatizácia, robotika a informatika* [online]. Copyright ©2010 ui42 [cit. 11.05.2020]. Dostupné z: https://www.atpjournal.sk/rubriky/exkluzivne-clanky/robotstudio-silny-nastroj-pre-online-i-offline-programovanie.html?page_id=2517

[6] RobotStudio, *RobotStudio Help* 2019.3 [cit 11.5.2020]

[7] Automatic sending station: QCX PTS102. *FLSmidth: Instalation, Operating & Maintanace Instruction*. 10.4. 2019, 103[cit. 11.5.2020]

[8] CX 8090: Embeded Pc for Ethernet. *TwinCAT: Documentation* [online]. 2.9. 2018, , 72 [cit. 11.5.2020]. Dostupné z: https://download.beckhoff.com/download/document/ipc/embedded-pc/embedded-pc-cx/cx8090_hwen.pdf

[9] TC2: Údržba Základní pojmy. *BECKHOFF: TwinCAT 2*. 2.9. 2018, , 15 [cit. 11.5.2020].

[10] GitHub - DenisFR/RSConnectGIOToSnap7: RobotStudio Smart Component to connect GI/GO to SIEMENS PLC using Snap7 library.. *The world's leading software development platform · GitHub* [online]. Copyright © 2020 GitHub, Inc. [cit. 11.05.2020]. Dostupné z: https://github.com/DenisFR/RSConnectGIOToSnap7

[11] Beckhoff Information System - English. *Beckhoff Information System - German* [online]. Dostupné z: https://infosys.beckhoff.com/english.php?content=../content/1033/tcadscomlib/html/tcadscomlib_intro.htm&id=

[12] TC3 TCP/IP: *Manual. BECKHOFF: TwinCAT3* [online]. 19.11.2019, , 93 [cit. 11.5.2020]. Dostupné z: https://library.e.abb.com/public/244a8a5c10ef8875c1257b4b0052193c/3HAC032104-001_revD_en.pdf

[13] Kruhová vyrovnávací paměť "buffer" pro mikrokontrolér ATmega(16/32) : Tajned - .NET and Embedded Development. *Tajned - .NET and Embedded Development* [online]. Copyright © http [cit. 1105.2020]. Dostupné z: http://www.tajned.cz/2015/08/kruhova-vyrovnavaci-pamet-buffer-pro-mikrokontroler-atmega1632/

[14] Circular Buffer Reference | *Arduino Libraries* | *MegunoLink. MegunoLink Pro* | *The swiss army knife for Arduino* [online]. Copyright © 2018 Number Eight Innovation. All rights reserved. [cit. 11.05.2020]. Dostupné z: https://www.megunolink.com/documentation/arduino-libraries/circular-buffer/